University of California
Santa Barbara

Digital Readout for Microwave Kinetic Inductance Detectors and Applications in High Time Resolution Astronomy

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Physics

by

Matthew James Strader

Committee in charge:

Professor Benjamin Mazin, Chair
Professor Omer Blaes
Professor Carl Gwinn

September 2016

The Dissertation of Matthew James Strader is approved.

Professor Omer Blaes

Professor Carl Gwinn

Professor Benjamin Mazin, Committee Chair

August 2016

Digital Readout for Microwave Kinetic Inductance Detectors and Applications in High Time Resolution Astronomy

Matthew James Strader

Acknowledgements

I owe tremendous gratitude to my advisor, Ben, for his years of guidance and encouragement. I also want to thank my labmates for making the lab a fun and enjoyable place to work, as well as their help in making this project possible. This work would also not have been possible without our collaborators, especially those at Fermilab. I thank them for their hard work and careful attention to detail. My family has been continually supportive through the six year ordeal of grad school. The friends I've made in grad school have had an important influence on my life and have kept me sane through the most difficult times. My heartfelt gratitude goes to them. The work in Chapter 8 was supported by NSF grant AST-1411613. The MKID detectors used in this work were developed under NASA grant NNX11AD55G.

Curriculum Vitæ

Matthew James Strader

Education

2016	Ph.D. in Physics (Expected), University of California, Santa Barbara
2013	M.A. in Physics, University of California, Santa Barbara
2010	B.S. in Physics (Applied Physics Option); B.S. in Computer Science (Highest Honors), California Statue University San Bernardino

Publications

"Search for optical pulsations in PSR J0337+1715," **M. J. Strader**, A. M. Archibald, S. R. Meeker, P. Szypryt, A. B. Walter, J. C. van Eyken, G. Ulbricht, C. Stoughton, B. Bumble, D. L. Kaplan, and B. A. Mazin 2016, MNRAS 459, 1.

"The ARCONS Pipeline: Data Reduction for MKID Arrays." J. C. van Eyken, **M. J. Strader**, A. B. Walter, S. R. Meeker, P. Szypryt, C. Stoughton, K. O'Brien, D. Marsden, N. K. Rice, Y. Lin, and B. A. Mazin 2015, ApJS, 219, 14.

"Direct Detection of SDSS J0926+3624 Orbital Expansion with ARCONS," P. Szypryt, G.E. Duggan, B.A. Mazin, S.R. Meeker, **M.J. Strader**, J.C. van Eyken, D. Marsden, K. OBrien, A.B. Walter, G. Ulbricht, T.A. Prince, C. Stoughton, and B. Bumble 2014, MNRAS, 439, 3.

"Excess Optical Enhancement Observed with ARCONS for Early Crab Giant Pulses," **M.J. Strader**, M.D. Johnson, B.A. Mazin, G.V. Spiro Jaeger, C.R. Gwinn, S.R. Meeker, P. Szypryt, J.C. van Eyken, D. Marsden, K. O'Brien, A.B. Walter, G. Ulbricht, C. Stoughton, B. Bumble 2013, ApJL 779, L12.

"ARCONS: A 2024 Pixel Optical through Near-IR Cryogenic Imaging Spectrophotometer," B. A. Mazin, S.R. Meeker, **M. J. Strader**, B. Bumble, K. OBrien, P. Szypryt, D. Marsden, J. C. van Eyken, G. E. Duggan, G. Ulbricht, A. B. Walter, C. Stoughton, and M. Johnson 2013, PASP, 123, 933.

Abstract

Digital Readout for Microwave Kinetic Inductance Detectors and Applications in High Time Resolution Astronomy

by

Matthew James Strader

This dissertation spans two topics relating to optical to near-infrared astronomical cameras built around Microwave Kinetic Inductance Detectors (MKIDs). The first topic is the development of a digital readout system for 10- to 30-kilopixel arrays of MKIDs. MKIDs are superconducting detectors that can detect individual photons with a wide range of wavelengths with high time resolution (2 µs) and low energy resolution. The advantage of MKIDs over other low temperature detectors with similar capabilities is that it is relatively straightforward to multiplex MKIDs into large arrays. All the complexity of readout is in room temperature electronics. This work discusses the implementation and programming of these electronics.

The second part of this work demonstrates the capabilities of the prototype optical and near-infrared MKID instrument with observations of pulsars. Detecting optical pulsations in these objects require high time resolution and low noise. The discovery of a correlation between the brightness of optical pulses from the Crab pulsar and the time of arrival of coincident giant radio pulses is presented. The search for optical pulses from a millisecond pulsar J0337+1715 is discussed along with a new upper limit on the brightness of its optical pulses.

Contents

List of Figures

Chapter 1

Introduction and Background

1.1 Microwave Kinetic Inductance Detectors

MKIDs are a relatively new low temperature detector (Day et al., 2003; Mazin, 2005; Gao, 2008). They are being developed for use at multiple wavelength ranges including, submm and mm (Janssen et al., 2013, Yates et al., 2011, Hubmayr et al., 2015, and others), near-infrared (NIR) through ultraviolet (UV; Mazin et al., 2013), and X-ray (Ulbricht et al., 2015; Miceli et al., 2014).

MKIDs enable the detection of individual photons with high time resolution (at the µs level) and with simultaneous energy resolution. MKIDs do not suffer from read noise or dark current in the way that CCDs (charge coupled devices), the standard detector for optical astronomy, do. The principle advantage of MKIDs over other low temperature detectors with attractive single pixel performance (Peacock et al., 1997; Irwin et al., 1996; Kraus et al., 1989; Enss, 2002) is that MKIDs are straightforward to scale up to large arrays through frequency domain multiplexing (McHugh et al., 2012; Duan, 2015; Swenson et al., 2012; van Rantwijk et al., 2016).

The operating principle of MKIDs is shown in Figure 1.1. For a careful analysis of the

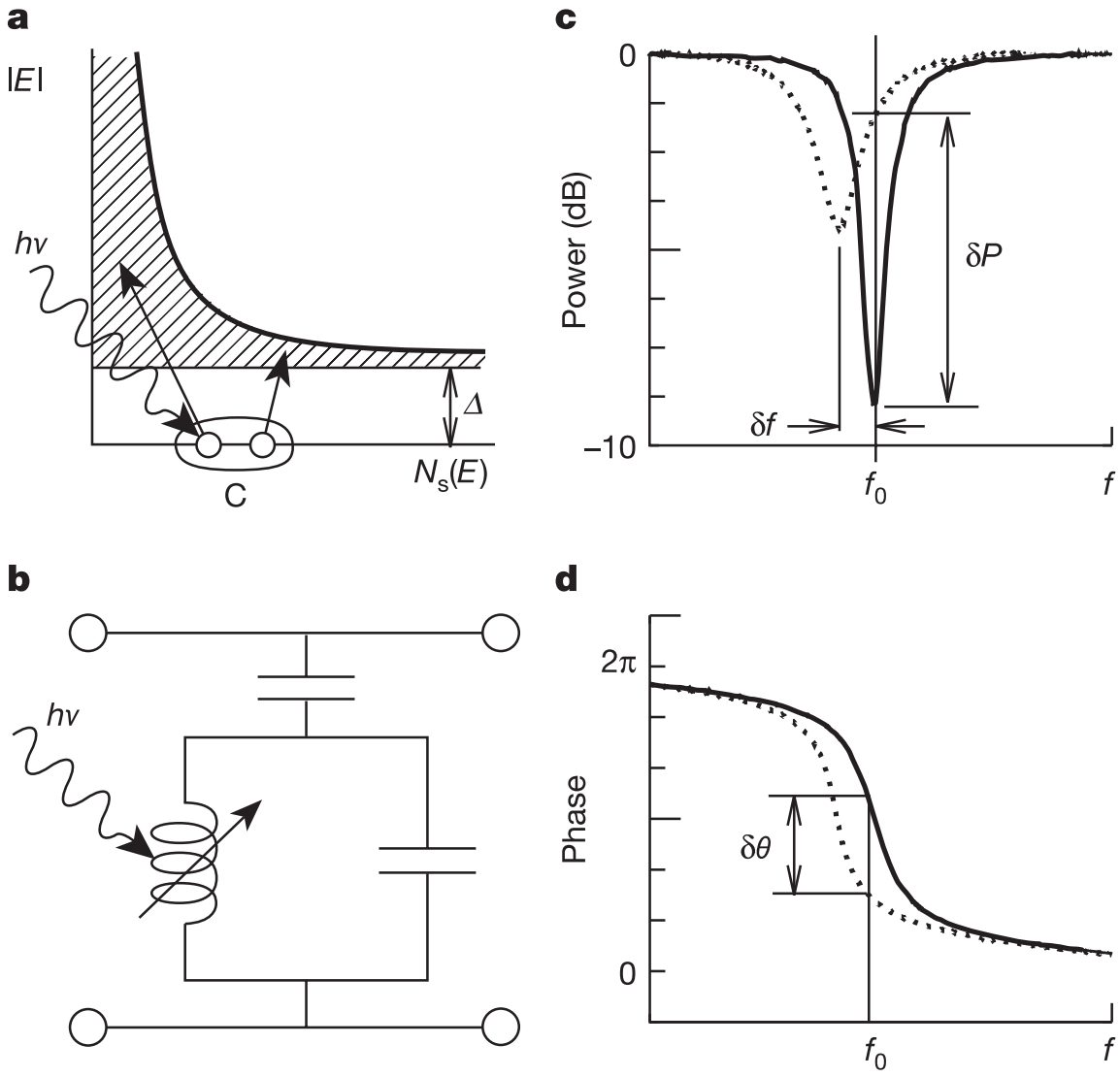

Figure 1.1: (a) MKIDs provide energy resolution because they have bandgaps much smaller than the energies of near-infrared to optical photons. When a photon strikes, it breaks Cooper pairs, producing quasiparticles in the conduction band. (b) The equivalent circuit diagram for an MKID is a simple LC resonator with a variable inductance. When a photon strikes the inductor, the inductance changes. (c) When the inductance changes, the resonant frequency of the resonator shifts down in frequency (from the solid line to the dotted line) and the Q of the the resonator decreases. (d) Along with this, the phase response changes. The phase of a probe tone passing through the MKID with frequency equal to the resonant frequency f_0 will shift $\delta\theta$, which is a function of the energy of the incident photon.

physics of MKIDs and derivations from Mattis-Bardeen theory that explain its operation, see Mazin (2005) and Gao (2008). The MKID acts as a simple LC (inductor-capacitor) circuit. An incident photon breaks Cooper pairs into quasiparticles in the superconducting film. This changes the surface inductance, which shifts the resonant frequency. The phase response also changes. A tone at the resonant frequency is passed through the MKID and monitored for changes in phase. When a photon is absorbed into the MKID inductor, the phase changes. The quasiparticles soon recombine and the surface impedance goes back to its original value. The sudden change followed by recombination manifests in phase as an exponential decay pulse (See Figure 2.7 for an example). The arrival time of the photon phase pulse can be measured precisely, providing high time resolution. Higher energy photons break more Cooper pairs, creating a larger phase pulse. So, measurement of the heights of pulses indicate photon energy. The energy resolution of this measurement could theoretically be as high as $R = E/\Delta E \sim 160$ at 4000 Å (Mazin, 2005), but at present, energy resolution measured in fabricated MKIDs is much lower.

The transmission of an MKID for frequencies far from the resonant frequency are near unity. This allows multiple tones to pass through an MKID, and only the tone at the resonant frequency will be affected by a photon hitting the MKID. This allows us to connect many MKIDs together on one feedline, with each MKID having a different resonant frequency. We create a comb of tones to pass through the feedline, with each tone corresponding to the resonant frequency for one MKID. Each MKID imprints an indication of incident photons only on its tone in the comb. In this way, thousands of MKIDs can be read out by a single wire (McHugh et al., 2012). All of the complexity of the readout then is at room temperature. After the frequency comb passes through the MKIDs all of these tones need to be separated from each other, so each can be monitored in parallel for changes in phase.

For a theoretical treatment of the noise expected for this kind of MKID readout, see Duan (2015). The goal in the readout design is for the noise contribution of the readout electronics to be less than that of the high electron mobility transistor (HEMT) amplifier, which amplifies signals on a feedline before they exit the cryostat.

In this work, the terms MKID, pixel, resonator, and (in the context of channelization firmware) channel are used interchangeably.

1.2 ARCONS

The Array Camera for Optical to Near-IR Spectrophotometry (ARCONS) was the first prototype instrument to make use of Microwave Kinetic Inductance Detectors (MKIDs; Mazin et al., 2013) for astronomical observing in UVOIR (ultra-violet, optical, and near-infrared) bands. ARCONS was optimized for the wavelenth range 4000-11000 Å. The operationg temperature was 110 mK. The energy resolution for ARCONS pixels was $R = E/\Delta E \sim 8$ at 4000 Å. The focal plane array consisted of 2024 MKIDs (46×44). The instrument's plate scale is 0.45"/pixel, so the field of view is 20"x20". The 2024 pixels were read out on two feedlines (1012 per feedline).

ARCONS was deployed at the Coudé focus of the 200" Hale Telescope at Palomar Observatory. It was a general purpose instrument used to observe a variety of objects, including optical pulsars (Strader et al., 2013, 2016) and compact binaries (Szypryt et al., 2014).

1.3 DARKNESS and MEC

The Dark-speckle Near-IR Energy-resolved Superconducting Spectrophotometer (DARKNESS) and the MKID Exoplanet Camera (MEC) represent the next generation of UVOIR

MKID instruments (Meeker et al., 2015). Both are specialized for direct imaging of exoplanets. DARKNESS has a 10,000 MKID array, designed for a wavelength range of 0.8-1.4 µm. The pixels are read out over five feedlines (2000 each). It was recently commissioned at the Cassegrain focus of the Hale Telescope behind the Stellar Double Coronagraph (SDC; Bottom et al., 2016) and the PALM-3000 adaptive optics system (Dekany et al., 2013).

MEC is still under development and will have a 20,000 MKID array. It will be integrated with the Subaru Coronagraphic Extreme Adaptive Optics (SCExAO) system (Jovanovic et al., 2015) at Subaru Telescope. The array will be read out on ten feedlines (2000 pixels each).

The digital readouts for these two systems will be nearly identical. The only difference will be that MEC requires twice as many electronic boards to read out twice as many pixels. In this work I will generally refer to the new readout as the DARKNESS readout or the DARKNESS firmware, even though it will be the same for both. DARKNESS was simply the one that was commissioned first.

1.4 Pulsars

Pulsars are rapidly rotating neutron stars with strong magnetic fields (typically $\sim 10^{12}$G for classical pulsars) (Lyne & Graham-Smith, 2012). Pulsars are divided into one of a few classes by their measured properties. This work focuses on the class known as rotation powered pulsars (RPPs), which convert rotational energy into electromagnetic radiation. This radiation has been observed in some pulsars in every wavelength range from radio to gamma rays (Mutel et al., 1974; Kuzmin et al., 2002; VERITAS Collaboration et al., 2011). Rotation powered pulsars are further divided into classical pulsars and millisecond or recycled pulsars, which have different properties due to a different

evolutionary histories. Classical pulsars are the remains of core collapse supernova (Lyne & Graham-Smith, 2012). Millisecond pulsars are understood to be classical pulsars that have been spun up by accreting material from a binary companion. See Lorimer (2005) for a review of millisecond pulsar and Mignani et al. (2011) for a review of observations of the few isolated pulsars with identified optical counterparts.

MKIDs are well suited to observations of optical pulsars compared to CCDs. The high time resolution makes it straightforward to detect pulsations. Unlike CCDs there is no added noise penalty for fast readout rates. In addition the wide wavelength sensitivity and low energy resolution of MKIDs allows for wavelength cuts to be done in post-processing. This lets us attempt to find the wavelengths with the highest signal-to-noise ratio (SNR) without prior knowledge. The intrinsic energy resolution allows us to extract phase-resolved spectra, which can add to our knowledge of pulsar emmision mechanism. The lack of dark current false counts in MKIDs allows us to attain a desired SNR faster, allowing us to search for faint optical pulsars.

Chapter 7 presents observations of the brightest optical RPP, the Crab Pulsar. Chapter 8 discusses observations of a millisecond pulsar PSR J0337+1715.

1.5 CASPER

The Collaboration for Astronomy Signal Processing and Electronics Research (CASPER) designs open-source hardware and software for use in radio astronomy applications. They designed the Reconfigurable Open Architecture Computing Hardware (ROACH), which is an electronics board with a field programmable gate array (FPGA), PowerPC processor, memory modules, and Z-DOK ports for mating with convertor boards (Parsons et al., 2006). The original ROACH had a Xilinx Virtex-5 FPGA. CASPER later designed a second generation board, the ROACH2, with a more powerful Virtex-6 FPGA. Along

with the hardware CASPER has developed an entire toolflow and set of libraries to aid in developing FPGA firmware and control software for these boards.

Although the original intent of the hardware was for use in radio astronomy instruments, it was made to be modular and its capabilities for digitial signal processing lend itself well to MKID readout.

1.6 Permissions and Attributions

1. Figure 1.1 is reprinted by permission from Macmillan Publishers Ltd: Nature 425, 817, copyright 2003.

2. Figure 2.1 is reprinted from McHugh et al. (2012) with the permission of AIP Publishing.

3. Figure 2.3 is an update of Figure 6 in McHugh et al. (2012).

4. The content of Part I is the result of a collaboration with Neelay Fruitwala, Alex Walter, Ted Zmuda, Kenneth Treptow, Neal Wilcer, Gustavo Cancelo, and Ben Mazin. The ADC/DAC boards and RF/IF boards were designed at Fermilab. The Distribution board was designed by Ben Mazin. The Virtex-7 firmware was principally written by Ted. Neelay and Ted wrote much of the C code used on the Virtex-7 MicroBlaze. Alex wrote much of the python graphical user interfaces to control the Virtex-6.

5. The content of Chapter 7 is the result of a collaboration with M.D. Johnson, B.A. Mazin, G.V. Spiro Jaeger, C.R. Gwinn, S.R. Meeker, P. Szypryt, J.C. van Eyken, D. Marsden, K. OBrien, A.B. Walter, G. Ulbricht, C. Stoughton, and B. Bumble, and has previously appeared in the Astrophysical Journal Letters (Strader et al.,

2013). It is reproduced here with the permission of the American Astronomical Society.

6. The content of Chapter 8 is the result of a collaboration with A. M. Archibald, S. R. Meeker, P. Szypryt, A. B. Walter, J.C. van Eyken, G. Ulbricht, C. Stoughton, B. Bumble, D. L. Kaplan, and B. A. Mazin, and has previously appeared in the Monthly Notices of the Royal Astronomical Society (Strader et al., 2016). It is reproduced here with the permission of Oxford University Press.

Part I

Digital Readout

Chapter 2

Principles and Algorithms

2.1 Introduction

The general algorithm used to read out large numbers of UVOIR MKIDs is described in McHugh et al. (2012). This work does not change any of the fundamental strategy of that explained in McHugh et al. (2012). Instead this work focuses on the impreve-ments necessary to scale the implementation of the strategy to high RF (radio frequency) sampling rates, larger numbers of pixels, and to other requirements of the new MKID cameras, DARKNESS and MEC. The ARCONS readout could process 256 pixels for each set of readout boards in 512 MHz of bandwidth. The DARKNESS readout must process 1024 pixels for each set of boards in 2 GHz of bandwidth. The algorithm that is tersely layed out in McHugh et al. (2012) is explained in detail below, as applied to the DARKNESS readout.

The readout uses I/Q data to represent complex waveforms. I (in-phase) represents the real part of a signal and Q (quadrature) represents the imaginary part. For a simple

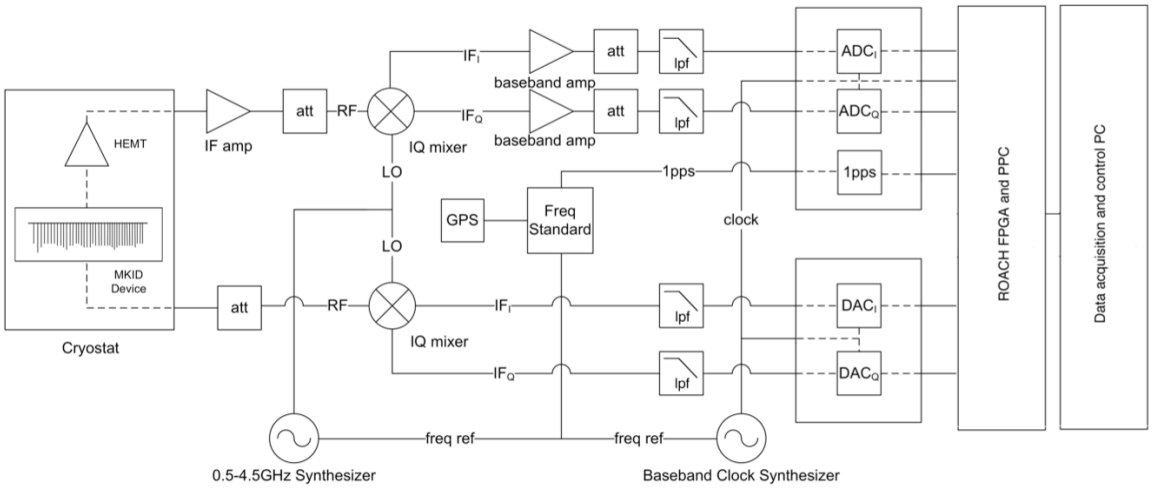

Figure 2.1: A block diagram showing the general readout strategy for the ARCONS readout. A comb of tones is generated, sent through the MKIDs, and then processed in FPGAs.

sinusoid with frequency f_{tone}, the I/Q signal can be written as

$$\begin{aligned} I_{\rm tone}(t) + iQ_{\rm tone}(t) &= e^{i2\pi f_{\rm tone}t} \\ &= cos(2\pi f_{\rm tone}t) + i\ sin(2\pi f_{\rm tone}t). \end{aligned} \tag{2.1}$$

The general strategy of the readout is mapped out in Figure 2.1. A frequency comb waveform is generated containing a number of tones that, once boosted to the 4 to 8 GHz range, match MKID resonant frequencies. The power of each tone in the comb is chosen to match the optimum readout power for its corresponding MKID. This waveform is mixed up in frequency by being multiplied by a local oscillator (LO) frequency in this range. Then the signal is sent through the MKIDs in the cryostat. When it returns to room temperature it is mixed back down and digitized by ADCs. FPGAs can then separate out the tones in the frequency comb. Once separated the FPGAs search the phase of the individual tones for indications that photons struck the corresponding MKIDs.

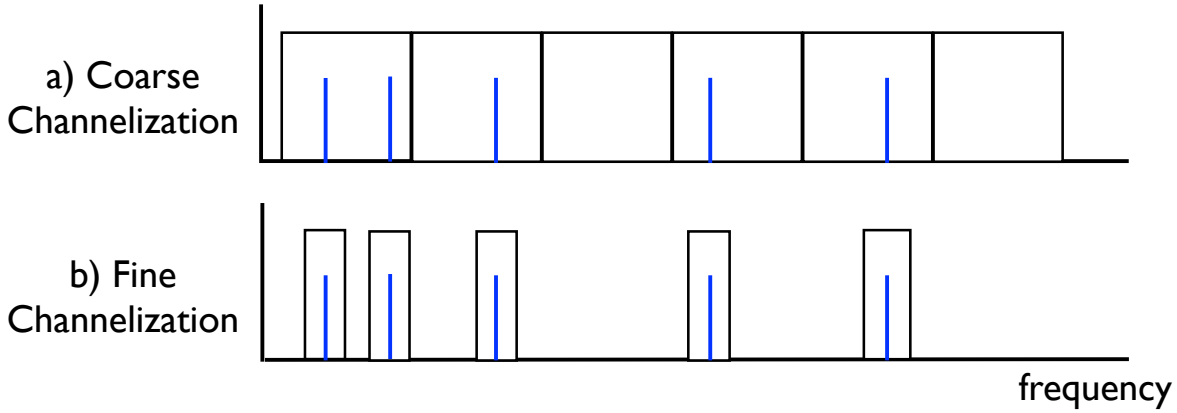

Figure 2.2: A cartoon showing tones being separated by channelization. Blue lines show the location of tone frequencies. (a) First coarse channelization by an FFT breaks the bandwidth into large equally spaced chunks. (b) Then the second stage makes smaller channels customized to the locations of tone frequencies.

2.2 Channelization Algorithm

To read out an MKID we generate a tone at the resonant frequency of the MKID, send it through the MKID, and look for changes in phase of the tone received back that indicate that the MKID was hit by a photon. Since we read out thousands of detectors in one feedline, we need to be able to take a frequency comb with many frequencies added together and divide them up into their individual tone frequencies, so each can be analyzed independently and in parallel. We use a Virtex-6 FPGA to do this task. Separation of the tone frequencies (called channelization) is done in two stages. The first stage takes a given bandwidth and divides it into a number of large equally sized and equally spaced chunks. Each chunk may happen to contain zero, one, or more tone frequencies in it. In the second stage, the large chunks are used to make narrow channels around where tone frequencies are (See Figures 2.2 and 2.3). In DARKNESS the initial large chunks are about 2 MHz wide and the narrow channels are 0.5 MHz.

For the first channelization stage, we would like to efficiently apply a series of bandpass filters to the I/Q timestream containing a frequency comb. For each filter we want to pass

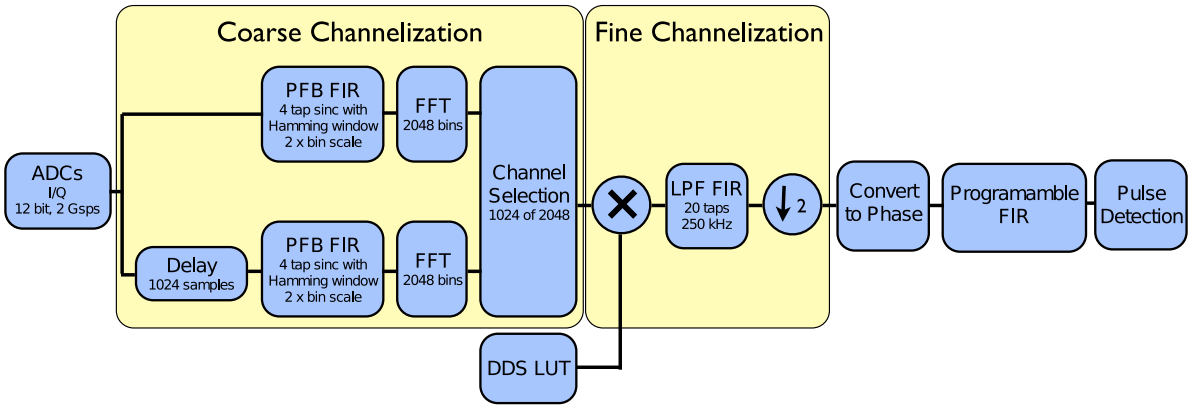

Figure 2.3: A block diagram of the processing done in firmware. The ADC digitizes a waveform containing all readout tones. The tones are separated by two stages of channelization. Once separated, the I/Q data are converted to phase. The phase is filtered and checked for photon pulses.

a certain chunk of frequencies completely undisturbed, while attenuating all frequencies outside of this chunk. We accomplish this with a polyphase filter bank (PFB), consisting of a finite impulse response (FIR) filter and a Fast Fourier Transform (FFT). Running the initial timestream containing all the readout tones through an N-point FFT will divide our frequency space into N equally sized and equally spaced bins. However, a standard FFT will not perfectly preserve a tone inside of its bin unless the tone happens to be exactly at its center. The frequency response of an FFT bin has the shape of a sinc function as shown in Fig 2.4. Frequencies that are not at the bin center would be attenuated to some degree. Also, the lobes in the frequency response function indicate that some frequencies far from the bin center may pass through the FFT with relatively little attenuation. To make each FFT bin more like a bandpass filter we first run the data through a PFB filter.

A four tap PFB multiplies N points from the incoming timestream with a sinc window function, splits the timestream into four segments and adds them together (Lyons, 2004). The sinc window changes the shape of the single bin frequency response to be more rectangular. Side lobes still exist in the frequency response function but they are suppressed

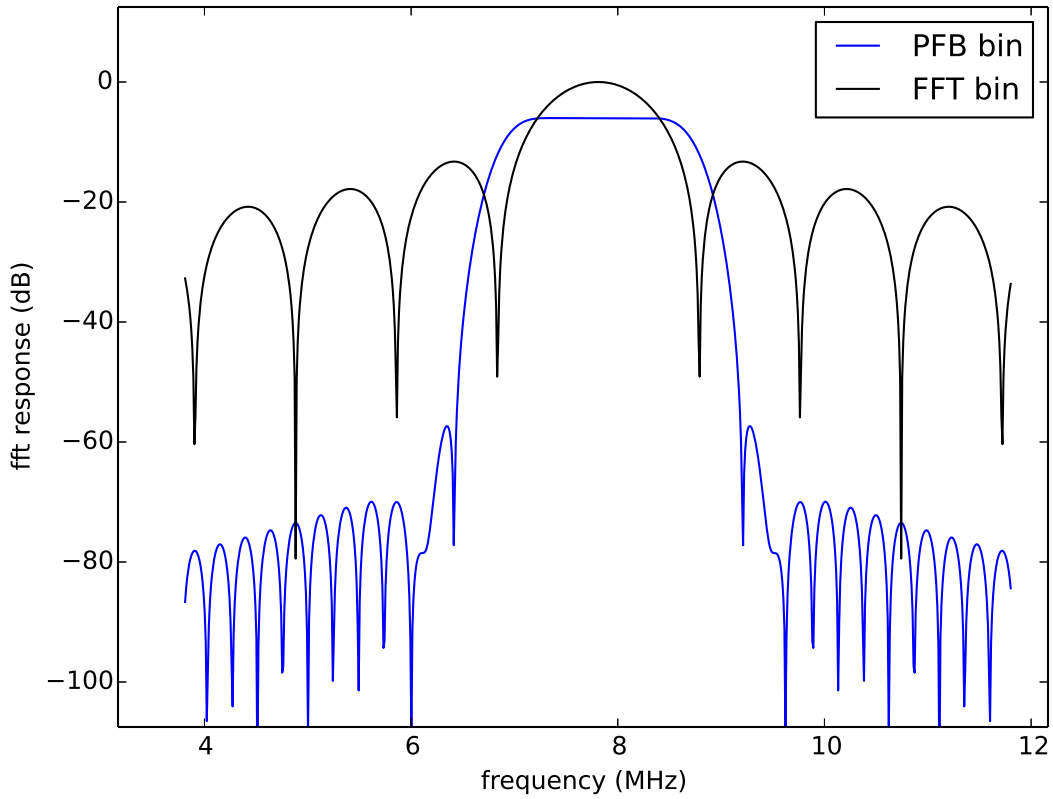

Figure 2.4: The single bin frequency response function for an ordinary FFT (black) and an FFT preceded by a PFB FIR (blue). A Hamming window was applied with the PFB FIR. The PFB flattens the response near the bin center and suppresses the response in sidelobes.

compared with those in the normal FFT response function. We use a hamming window to maximize the suppression of the first side band. We can also customize the width of the passband of the bin by adjusting the width of the sinc window we use. We choose to broaden the frequency response so that the passband of neighboring FFT bins overlap (See Figure 2.5). We do this so that no matter what frequency a tone happens to be at, there is at least one FFT bin that passes it through with nearly zero attenuation, along with at least 0.5 MHz of bandwidth around it. This will provide enough room for the final 0.5 MHz channel around the tone, once we perform the second stage of channelization.

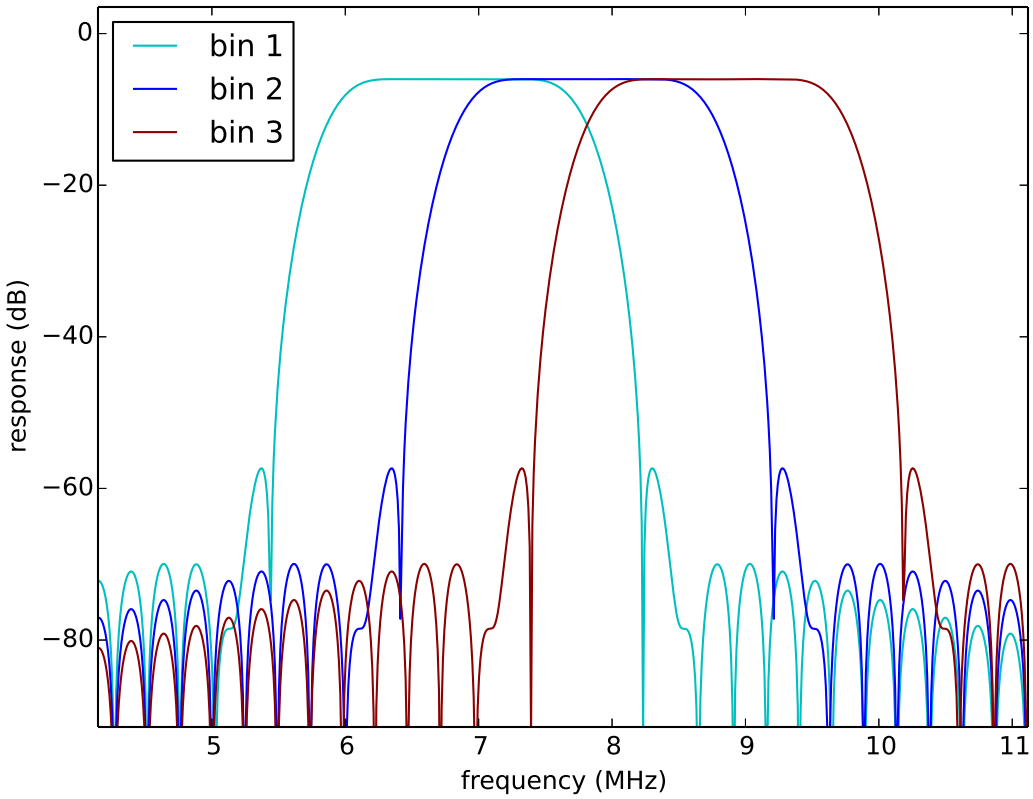

Figure 2.5: The single bin frequency response for three neighboring FFT bins. The PFB FIR has been adjusted to widen the flat area of frequency response. No matter where a tone frequency may be, there is at least one FFT bin that will pass it with minimal attenuation.

In the DARKNESS firmware we use a 2048 point complex FFT. The I/Q input timestream is sampled by the ADC at $f_s = 2\,\text{GHz}$. Using I/Q data allows us to represent complex sinusoids with both positive and negative frequencies. The 2 GHz sample rate allows frequencies with absolute value less than the Nyquist frequency $f_{\text{Nyq}} = \frac{f_s}{2} = 1\,\text{GHz}$ to be distinguished (Lyons, 2004). So, the input timestream has a total bandwidth of 2 GHz (representing frequencies from -1 GHz to +1 GHz). After passing through a PFB FIR and FFT 2048 points at a time, we receive values for 2048 bins, each of which becomes

a new timestream, time multiplexed together. The bin center frequencies include

$$\begin{aligned} f_{\text{bin center}} &= [-\frac{1}{2}, \quad ..., -\frac{1}{N}, \quad 0, \frac{1}{N}, \quad ..., \left(\frac{1}{2} - \frac{1}{N}\right)] && 2\,\text{GHz}. \\ &= [-\frac{1}{2}, \quad ..., -\frac{1}{2048}, \quad 0, \frac{1}{2048}, \quad ..., \left(\frac{1}{2} - \frac{1}{2048}\right)] && 2\,\text{GHz}. \\ &= [-1000, \quad ..., -0.98, \quad 0, 0.98, \quad ..., 999.02] && \text{MHz}. \end{aligned}$$

When a tone frequency is exactly at the center of an FFT frequency bin, the bin I and Q values are constant over time. If a tone frequency is not at a bin center, the magnitude of the complex signal $\sqrt{I^2 + Q^2}$ will remain constant over time but the bin I and Q values will oscillate with the difference of frequencies

$$f_{\text{bin,osc}} = f_{\text{tone}} - f_{\text{bin center}}. \tag{2.2}$$

With a single FFT, each bin is sampled once per 2048 input data points. So, the timestream of a particular bin is sampled at

$$\begin{aligned} f_{s,bin} &= \frac{f_s}{N} \\ &= \frac{2\,\text{GHz}}{2048} \\ &= 0.977\,\text{MHz} \\ &\approx 1\,\text{MHz}. \end{aligned}$$

(Hereafter, multiples of $\frac{2\,\text{GHz}}{2048}$ will simply be approximated as multiples of 1 MHz.)

The spacing between FFT bin centers is 1 MHz, and ordinarily, the passband of a PFB conditioned FFT bin would match this. However, we have broadened the passband

to 2 MHz, passing frequences that satisfy

$$|f_{tone} - f_{bin\ center}| < 1\,\text{MHz}.$$

Assuming we use a single FFT with an implied bin sampling rate of 1 MHz, the Nyquist frequency would be half this, 0.5 MHz. This means that a tone with a frequency satisfying

$$0.5\,\text{MHz} < |f_{tone} - f_{bin\ center}| < 1\,\text{MHz}$$

would appear in the FFT bin timestream unattenuated but would be aliased into the range

$$|f_{tone} - f_{bin\ center}| < 0.5\,\text{MHz}.$$

This could interefere with other tones in the same bin.

To solve this, we need to double the sample rate of each FFT bin. We do this by performing two FFTs in parallel, where the input to one of the FFTs is timeshifted by $\frac{N}{2} = 1024$ samples relative to the first. We also adjust the phase of bin values exiting the second FFT (by multiplying odd bins by -1) to make them match up with the first such that we can interleave the values for a particular bin from both FFTs and make a single coherent timestream sampled at 2 MHz. At this point we have successfully divided our frequency space into equally spaced overlapping chunks of frequency (as in Figure 2.2a).

For the second stage of channelization we use a digital down conversion (DDC) approach (See Figure 2.6). For each tone frequency we multiply an FFT bin timestream with a complex sinusoid with frequency matching that in Eq (2.2). We call this complex sinusoid the DDS (direct digital synthesizer), after the hardware that would be used for this task in an analog homodyne system. This shifts the frequency of interest to zero.

We then a apply a low pass filter with a cutoff frequency of $f_c = 0.25\,\mathrm{MHz}$ to both I and Q. This will kill any other readout tones in the same FFT bin that we are not at the moment interested in, provided the readout tones have a minimum spacing of 0.5 MHz. This also preserves the frequency space immediately around the tone frequency, so we can watch for changes in phase as fast as 2 µs without smoothing them away. This will allow us to detect photons with adequate time resolution. For FFT bins containing two (or more) tone frequencies we make two (or more) copies of the bin timestream and multiply with different DDS frequencies. In one bin copy the first tone is moved to zero frequency and the second is killed by the low pass filter. In another copy the second tone is moved to zero frequency and the first is killed by the low pass filter. We also throw away the timestreams for any bins that do not contain tone frequencies. All of this can be decided while setting up the readout, because we already know the tone frequencies and what bins they will fall in. In this process we go from 2048 FFT bins to 1024 frequency chunks of width 0.5 MHz, each containing exactly one tone frequency (which has been shifted to zero frequency) as in Figure 2.2b. At this point we dub these frequency chunks "channels."

Once the low pass filter has narrowed the bandwidth of a channel to 0.5 MHz, we can safely reduce the sample rate of the channel without danger of aliasing larger frequencies into our band. We could downsample to 0.5 MHz, but instead we downsample to 1 MHz to simplify the firmware (See Section 4.3.2). Throughout this process, we continue to use complex I/Q data.

2.2.1 Comparison to other approaches

The above approach involving both an FFT and multiple parrallel DDCs is complicated. One should consider the tradeoffs in using other simpler approaches. One

approach would be to dispense with the DDCs, and simply use a very large FFT. This is the tactic taken for the readout of some other MKID cameras (Bourrion et al., 2012; Duan, 2015; Swenson et al., 2012; van Rantwijk et al., 2016). Increasing the number of points in the FFT decreases the spacing between bin centers. With a sufficiently large FFT, there would be enough bins with small enough spacing between them that for any arbitrary tone frequency, there will be a bin center frequently close enough and narrow enough that the bin can be used as the channel for that tone without needing to further process it to filter out other tones. In addition, a PFB is not needed before the FFT to change the frequency response The tone will be close enough to the bin center that it will be transmitted with minimal attenuation, and the bin response frequency (including it's side lobes) is narrow enough that no other tones will leak into the channel with any noticeable strength. This approach is much simpler than ours and it is highly efficient in FPGA resources. However, it sacrifices time resolution to gain this simplicity. The final channels are much narrower than ours (For NIKA2 the highest final readout rate is 1272 Hz; van Rantwijk et al., 2016). This approach can only be used when the application does not require detection and timing of individual photons. The MKID cameras mentioned above are optimized for submm or mm wavelength applications in which the measured signal is integrated flux, not individual photons.

Another potential approach would be to dispense with the FFT and only use DDCs. For this we would duplicate the input timestream by the number of channels we wish to have. For each channel, we would need a DDS signal with frequency matched to the appropriate tone frequency in the input timestream to multiply with, shifting this tone frequency to zero before applying a low pass filter. In this way we could isolate each channel just as well as in the above scheme. The difference is the number of FPGA and memory resources necessary. To only use DDCs we would need a LUT containing 1024 complex DDS signals each sampled at 2 GHz, and all of these need to be multiplied with

the input I/Q timestream in parallel. In contrast, the two stage scheme requires a LUT containing 1024 DDS signals sampled at 2 MHz each. These DDS signals are multiplied with the FFT bin timestreams such that it is mostly time multiplexed, so the multiplies do not have to all be done in parallel.

Combining the considerations from these two extremes, we need to use an FFT with few enough points that we will be able to make final channel bandwidths large enough to achieve our desired time resolution, but we also need an FFT with enough points to minimize the FPGA resources needed to implement it as well as the DDSs, so that it will fit in the FPGAs in our boards. For input timestreams sampled at 2 GHz containing 1024 readout tones and a desired time resolution of 2 µs we settled on a 2048 point FFT.

2.3 Photon Detection

Once the tones in each channel have been separated by the channelization scheme, it is time to search the information in each channel for signs of photons striking the corresponding MKID and changing the surface impedance. This manifests as a negative exponential pulse in the phase timestream of the readout tone (See Figure 2.7). At the end of the second stage of channelization we have timestreams of I and Q values from time multiplexed channels. We convert the I and Q to phase by calculating

$$\phi = \texttt{arctan}\left(\frac{Q - Q_{\text{center}}}{I - I_{\text{center}}}\right) \tag{2.3}$$

where $(I_{\text{center}}, Q_{\text{center}})$ is the center of the MKID's resonance loop in the I/Q plane. The amplitude

$$A = \sqrt{(I - I_{\text{center}})^2 + (Q - Q_{\text{center}})^2} \tag{2.4}$$

will also show changes in the MKID surface impedance, but in practice the SNR is several times higher in phase. The amplitude signal is also more sensitive to the detection of the resonator I/Q loop center, which can be difficult to detect reliably when the transmission of the feedline changes with frequency. For these reasons it is simpler to convert to phase and throw away the amplitude information, without losing much signal in the process.

2.3.1 Optimal Filtering and Baseline Subtraction

The raw phase timestream can be searched for photon pulses, but noise increases the uncertainty in photon arrival time and pulse height, which in turn hurts the energy resolution. Before searching for pulses, the phase timestreams are passed through an FIR filter. The FIR coefficients are customized to each channel to maximize the SNR of photon pulses (See Figure 2.8). In the ARCONS filter there were only enough resources for a 26 tap programmable filter. This small number of taps limited the potential effectiveness of the filter. Only 26 µs of phase data is covered with this filter, while photon pulse lifetimes range from about 30 µs to 50 µs. At this timescale it is reasonable to assume that the noise can be well modeled as simple white noise. A matched filter maximizes the SNR given an exponential pulse and a known noise spectrum (Lyons, 2004). Also, a matched filter is applied in the time domain and can be applied to data in real time in FPGA firmware, unlike other types of filters that are also optimal in some metric of SNR (e.g. Wiener filter; Lyons, 2004). The matched filter coefficients g are given by

$$g = \frac{C^{-1}v}{v^T C^{-1} v} \tag{2.5}$$

where C is the covariance matrix derived from the known noise spectrum and v is the pulse template formatted as a column vector . For simple white noise the covariance matrix is simply the identity matrix and the filter coefficients resolve to be the same as

the filter template (Lyons, 2004).

Filtered data is computed as the correlation of the raw data with the filter coefficients. After filtering with a matched filter, the exponential pulse peaks still show as peaks, but they are smoother and more symmetric. Both the rise time and fall time are visible (See Figure 2.9).

To generate the pulse template we record phase timestreams of a large number of photon pulses of intermediate energy (600 nm for ARCONS). We line up these pulses and take the average to generate a pulse template. Often, these templates would still contain noise, so we would fit the template with an exponential decay function. This cleaner template could then be multiplied by the inverse covariance matrix, or more often used directly as filter coefficients with an appropriate normalization (using the fact that in the ideal case C^{-1} is the identity matrix).

With the additional resources of the Virtex-6, the DARKNESS firmware is compiled with 50 taps in the programmable filters, and with some optimization of FPGA resources might be extended to 100 taps. This gives us more room to use better filters that capture behavior on slightly longer timescales to improve energy resolution. One consideration in improving filtering is pulse pile-up. If two photons arrive at a pixel relatively close in time, in phase the second photon pulse will ride on the decaying exponential tail of the first one. If this is not taken into account, the second pulse will appear to originate from a higher energy photon than it actually did. Alpert et al. (2013) have formulated a generalization of the matched filter to address this sort of nuisance factor. They applied their specialized matched filters to pulses from X-ray Transition Edge Sensors (TES). A matrix is built with the first column being the pulse template that we want to maximize response to, and other columns are nuisance vectors that we want to minimize response

to. The filter coefficents that take k nuisance vectors into account are described by

$$g = C^{-1}V(V^T R^{-1} V)^{-1} e_1 \tag{2.6}$$

where V is the matrix containing the pulse template and nuisance vectors and e_1 is a unit vector $(1, 0, \cdots, 0)^T$ with length equal to $k + 1$.

To minimize the effect of pulse pile-up, one nuisance vector is added with the exponential decay rate of pulse tails.

Another nuisance is the phase baseline. If the baseline slowly shifts due to low frequency noise, we do not want photon pulses to be tagged as higher or lower energy due to this. In the ARCONS firmware, this was handled by subtracting the baseline before checking for pulse triggers. The baseline was found using a low pass state variable filter (SVF; Chamberlin, 1980) described by the diagram in Figure 2.10. The constants are determined as

$$k_f = 2 \ \mathtt{sin}(\pi \frac{f_c}{f_s}) \tag{2.7}$$

$$k_q = \frac{1}{Q} \tag{2.8}$$

where f_c is the desired cutoff frequency, f_s is the sample rate of the phase (1 MHz), and Q is a quality factor that determines the shape of the frequency response near the cuttoff frequency.

This type of filter was chosen because it takes relatively few FPGA resources to compute, and it can achieve very low frequency cutoffs without needing high precision coefficients. For ARCONS the cutoff frequency chosen was $f_c = 200$ Hz, with the hope that it would cut down on 60 Hz line noise. This baseline subtraction in firmware before pulse detection had an important side effect. The low pass filtered baseline is somewhat

sensitive to phase pulses. When a pulse occurs, the computed baseline temporarily shifts and then recovers. The lower the cutoff frequency is, the less sensitive the filter is to pulses, but the shifts will last longer. Because the pulse trigger threshold (discussed in Section 2.3.2) follows the baseline, these baseline shifts may prevent photon pulses from being detected. In particular after a large pulse, from a higher energy photon, low energy photons are not be detected for a time, and the ones that are detected are tagged as lower energy than they really are. This effectively creates an energy dependent deadtime. This potentially causes some strange artifacts in the ARCONS data that are difficult to compensate for. The best solution for this technique seems to be to use a very low cutoff frequency (around 20 Hz) to minimize these temporary baseline shifts and to not try to remove 60 Hz noise with this technique.

Another possible solution is to add a nuisance vector for a DC baseline in computing the generalized matched filters discussed above (See Figure 2.11). Preliminary tests on ARCONS phase data show that it is most effective to use both techniques together.

2.3.2 Peak Finding Conditions

After filtering and subtracting a baseline, the firmware checks for peaks in the filtered phase. There are several conditions that must be met in order for a pulse to be tagged as a photon.

First, it must be a negative peak, which is seen as a change in the discrete derivative of the phase from negative to postive. In the original ARCONS firmware this condition was met whenever the derivative was negative for one sample and positive for two consecutive samples. It was found, however, that if there was some high frequency noise in the filtered noise, this condition would be met multiple times for a single photon pulse, as there would be multiple bumps in phase riding on the larger exponential pulse. Sometimes

these bumps would erroneously be tagged as multiple photons, though in ARCONS the deadtime condition usually excluded them (discussed below). The next few conditions will preclude many of these extra pulses from being recorded as photons, but not always. So, this condition was made more strict. If there are multiple bumps on a pulse, we try to catch the first one (usually the most negative) to be tagged as a photon. The condition checks that for such a pulse, the derivative is negative for at least nine out of ten consecutive samples, and then positive for two consecutive samples. These numbers were chosen while analyzing real phase timestreams to minimize the number of false photon tags while also not missing virtually any real photons. A steep low pass filter would also smooth out the bumps and prevent false positives, but these strict trigger conditions require less FPGA resources than adding taps to the I/Q low pass filters.

The second condition is that the negative peak exceed a theshold, such that

$$(\phi_{\text{peak}} - \phi_{\text{baseline}}) < \phi_{\text{threshold}} \tag{2.9}$$

The threshold is customized for each pixel. This may be adjusted depending on the typical pulse size on a particular device, but the general rule of thumb is to set the threshold at 4 standard deviations below 0°. For mostly white gaussian phase noise this determines the percent of random phase fluctuations that meet this condition. A phase histogram of a pixel's triggered pulses shows the end of a gaussian tail cut off at the value of the trigger. The false triggers from noise can be cut out in post-processing if the gaussian noise tail is well separated from real photon triggers in the phase histogram. This is easier for pixels with higher energy resolution. If they cannot be distinguished we have to accept a certain proportion of false photon triggers in the data, particularly in the lowest photon energy bins.

The third condition for a pulse trigger is that a peak not occur too soon after a

preceding pulse trigger. In the ARCONS firmware the deadtime after a trigger was 100 µs. The purpose is to cut down on false triggers from noise peaks in a pulse's exponential tail. Depending on the count rate in a pixel we lose a proportion of good photons to this condition. To minimize this the deadtime has been reduced to 10 µs in the DARKNESS firmware. The potential false triggers after this 10 µs are reduced by the stricter peak condition described above.

The final trigger condition is a count rate limit. Problems can arise in the readout if many pixels simultaneously begin to trigger too often, such as if a very bright object was imaged with the pixel array. Too many pulse detections in a given time period would overload the DAQ system. In the ARCONS firmware, detected photons are stored in a small circular memory buffer to be read off by a C program running on the Roach1 PowerPC processor. If photons are written to the buffer too quickly, new photons begin to overwrite old photons before they are read off, so the latter are never recorded to disk. In the DARKNESS firmware, photon data is instead sent directly to the main DAQ computer by ethernet. The ethernet core also has buffers that can overflow, which locks up the ethernet core making it necessary to restart the core in order to continue. In both systems, a C program running on the main DAQ computer would receive photon data and write it to hard disk, and if the photon rate is too high, it would receive data faster than it could be written and the receive buffers would overflow.

Besides a bright object, "hot" pixels can also generate high count rates. In ARCONS TiN devices, we find that pixels can randomly, significantly, and temporarily increase their phase noise, producing a large number (hundreds to thousands per second) of noise peaks beyond the thresholds set for those pixels when they were less noisy. These pixels suddenly produce many false photon detections. The condition lasts between a few seconds to a few minutes and then subsides.

Whichever way that high count rates are produced, we want to prevent high count

rates in some pixels from causing data to be lost in other pixels. So, we impose a limit on the pulse detections that can be found in a particular pixel per second. In ARCONS, this limit existed only in the C code on the DAQ computer that collected and wrote data to disk. For DARKNESS, a limit of 2500 counts/pixel/s is applied in the firmware.

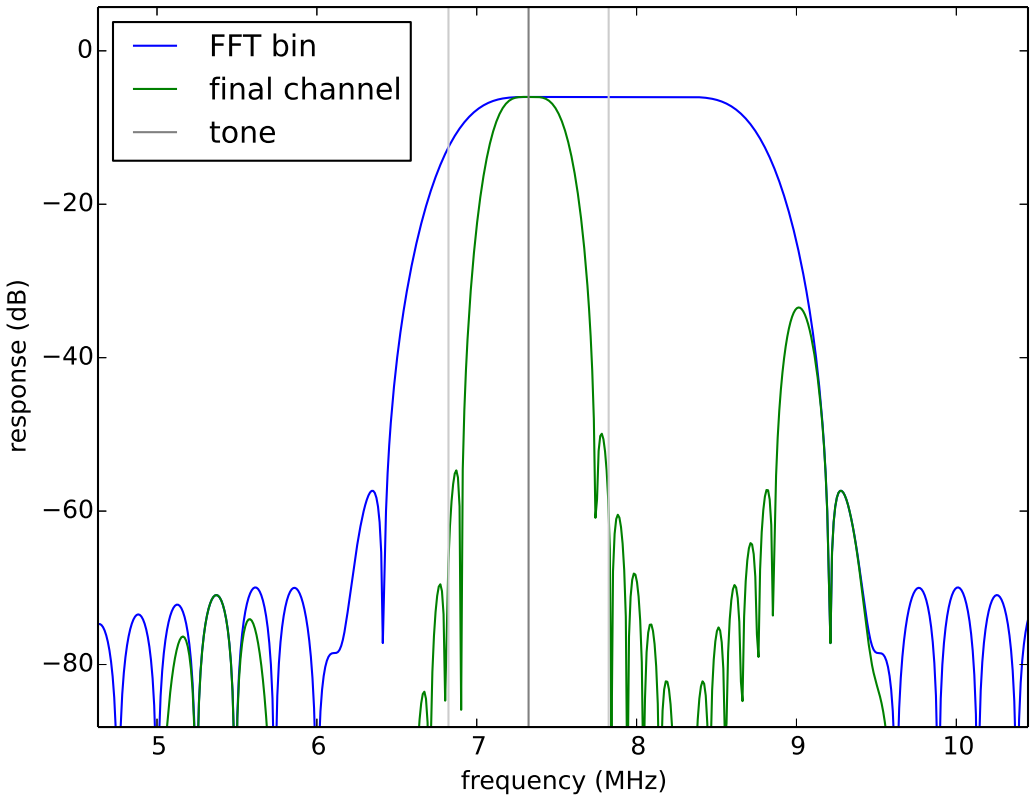

Figure 2.6: The frequency response function for one bin after coarse channelization (blue) and one channel after fine channelization (green). The final channel is centered on the location of a tone frequency (dark grey). Other tone frequencies may be present in the FFT bin (light grey), but will be attenuated enough in the final channel so as to not interefere with the dark grey tone. Another channel can be made with the same FFT bin around the right (higher frequency) light grey tone that excludes the dark grey tone.

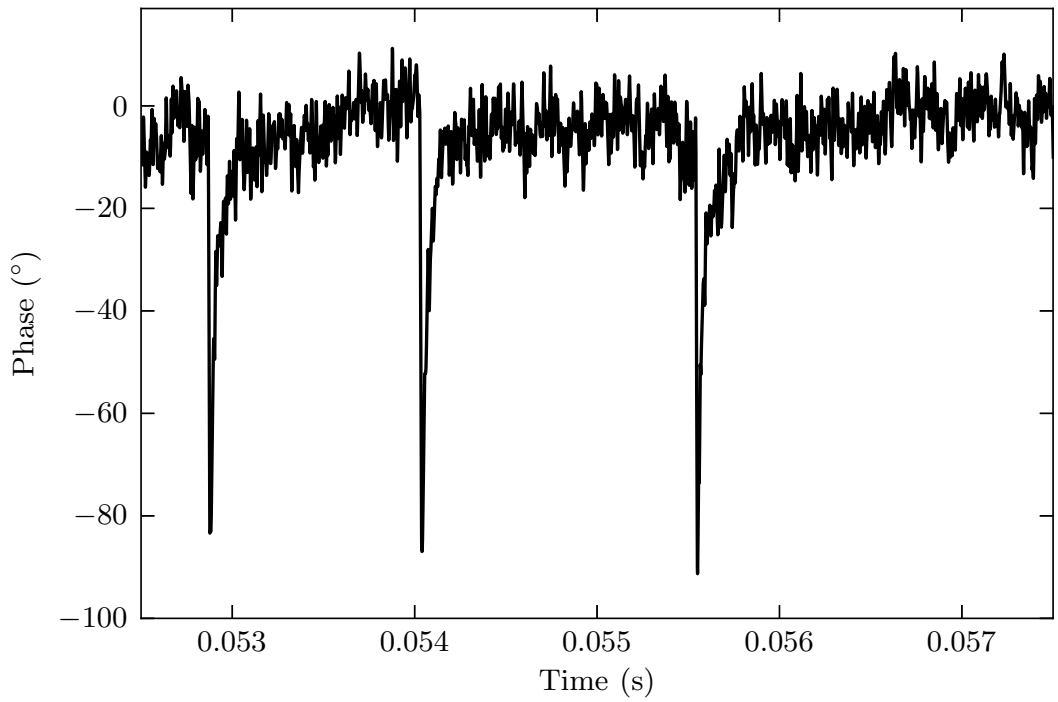

Figure 2.7: Three pulses in the phase timestream of one MKID indicate when three photons hit the device. The phase was passed through a 250 kHz low pass filter instead of a matched filter.

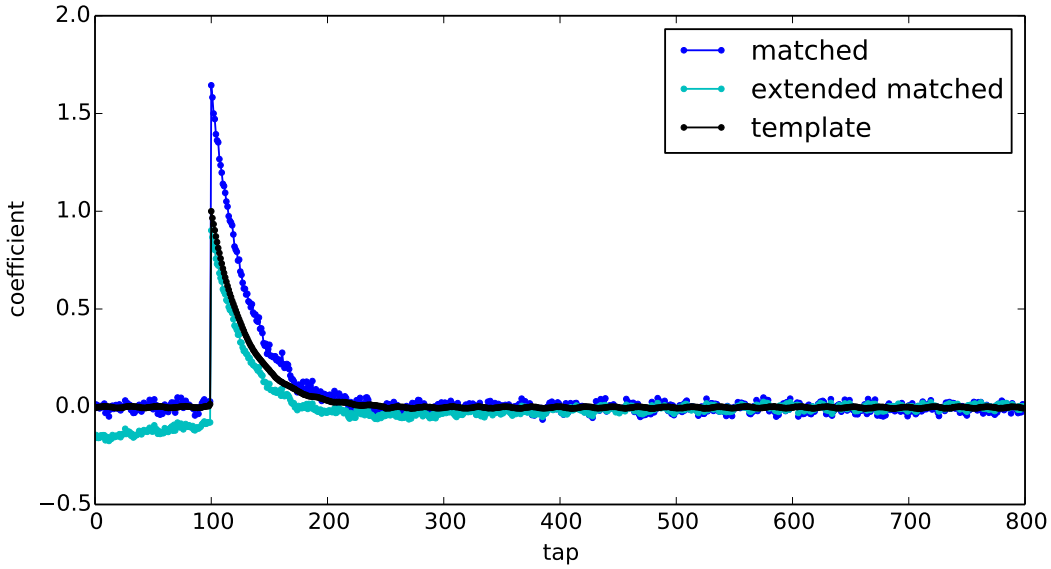

Figure 2.8: Coefficients for three possible FIR filters for peak detection constructed from simulated phase pulses with white noise with an additional low frequency and one high frequency added. These filters have 800 taps, which is far more than could be used in firmware. Black shows a simple exponential template filter fit to have the same decay time as average photon pulses. The matched filter incorporates information about the phase noise spectrum and tries to maximize the SNR for pulses shaped like the template. The extended matched filter is made orthogonal to two nuisance vectors, one for the low frequency baseline, and one for pulses riding on exponential tails from previous pulses (with folding time 200 µs).

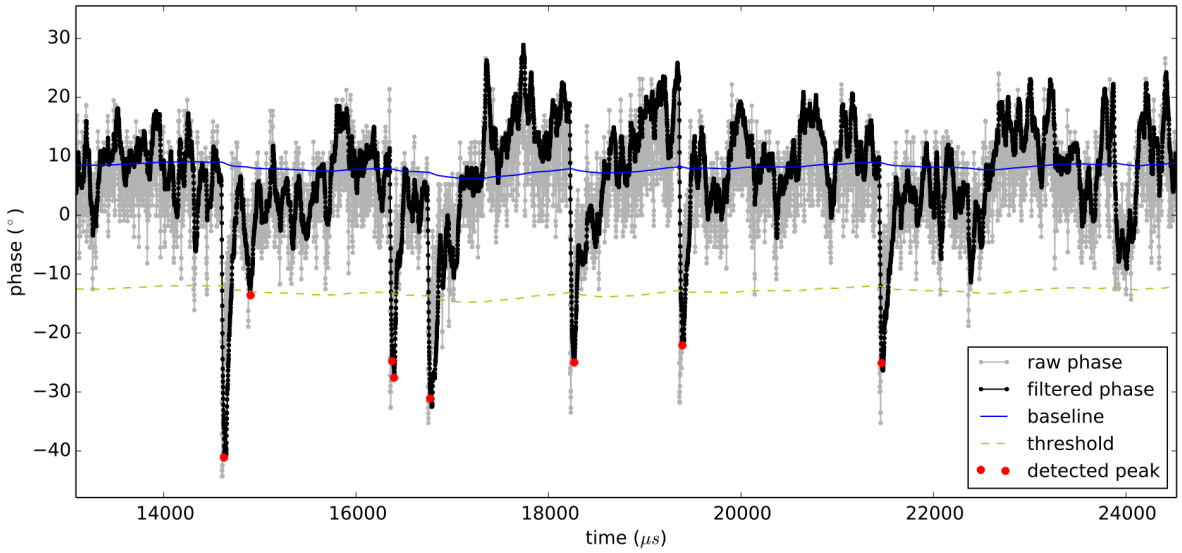

Figure 2.9: A phase timestream from an ARCONS pixel. The light gray shows raw unfiltered phase. The black line shows the result of setting the programmable filter to a 50 tap exponential template (with a 30 µs folding time). The blue line shows the baseline computed with an SVF filter ($f_{cutoff} = 20\,\text{Hz}$). The yellow dashed line shows how far down a phase peak must be to be detected as a photon. Red circles highlight phase points that meet all trigger conditions. One point (at $t = 14\,900\,\mu\text{s}$) is a noise trigger. This could be recognized and cut in post-processing by how close it is to the threshold. The pulse at $t = 16\,400\,\mu\text{s}$ is detected twice due to noise at the peak. This might be recognized and cut in post-processing by how close in time and phase these triggers are. Alternatively, better filtering may improve the noise.

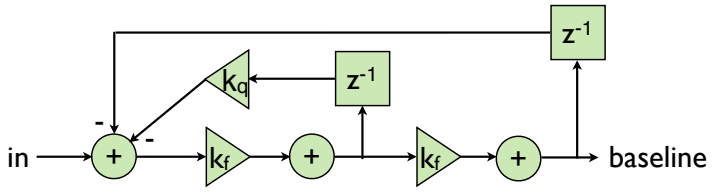

Figure 2.10: The block diagram for the digital state value filter used to find the phase baseline of each channel in firmware.

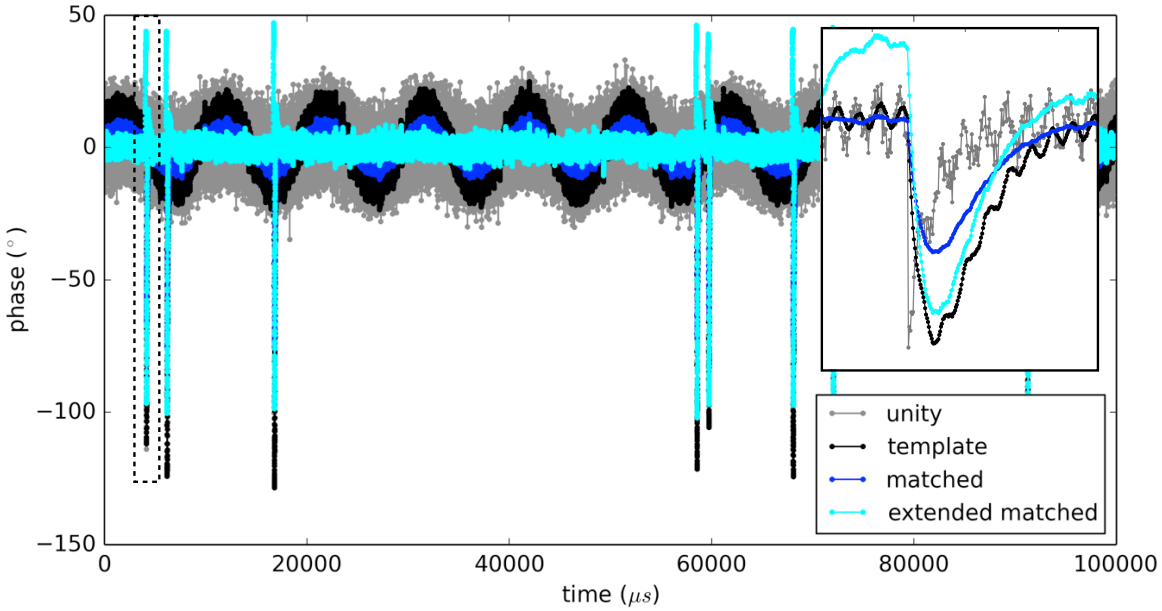

Figure 2.11: A simulated phase timestream with pulses for one pixel with various 800 tap filters. The simulated phase has white noise, and two nuisance sine waves added (one very low frequency and one very high frequency). The inset shows the phase pulse in the dashed line box. The gray is unfiltered raw phase. The black uses a simple exponential template filter. The extended matched filter effectively removes both the low frequency and the high frequency noise.

Chapter 3

Hardware

3.1 Introduction

In this chapter I cover all of the electronics boards needed to implement the channelization and pulse detection covered in the previous chapter. The three types of boards involved are the ROACH2, the ADC/DAC board and the RF/IF board. I call a set of these three boards one readout unit (See Figures 3.1 and 3.2). In the ARCONS readout, eight ROACH boards along with eight ADC/DAC boards are used to read out up to a total of 2048 MKIDs. Each ROACH board reads out 256 MKIDs in 512 MHz of bandwidth. In the DARKNESS readout ten ROACH2 boards, each connected to an ADC/DAC board, are used to read out up to 10,240 MKIDs. Each readout unit reads out 1024 MKIDs in 2 GHz of bandwidth.

FPGA boards are used in the readout rather than graphical processing units (GPUs) or similar processing hardware, because FPGAs are good at highly parallelized processing in which results can be obtained in real time at a reliable period. GPUs can also process data with high parallelization but they are not designed to produce results with strict regularity. For GPUs latencies between results may vary. In an FPGA we can be sure to

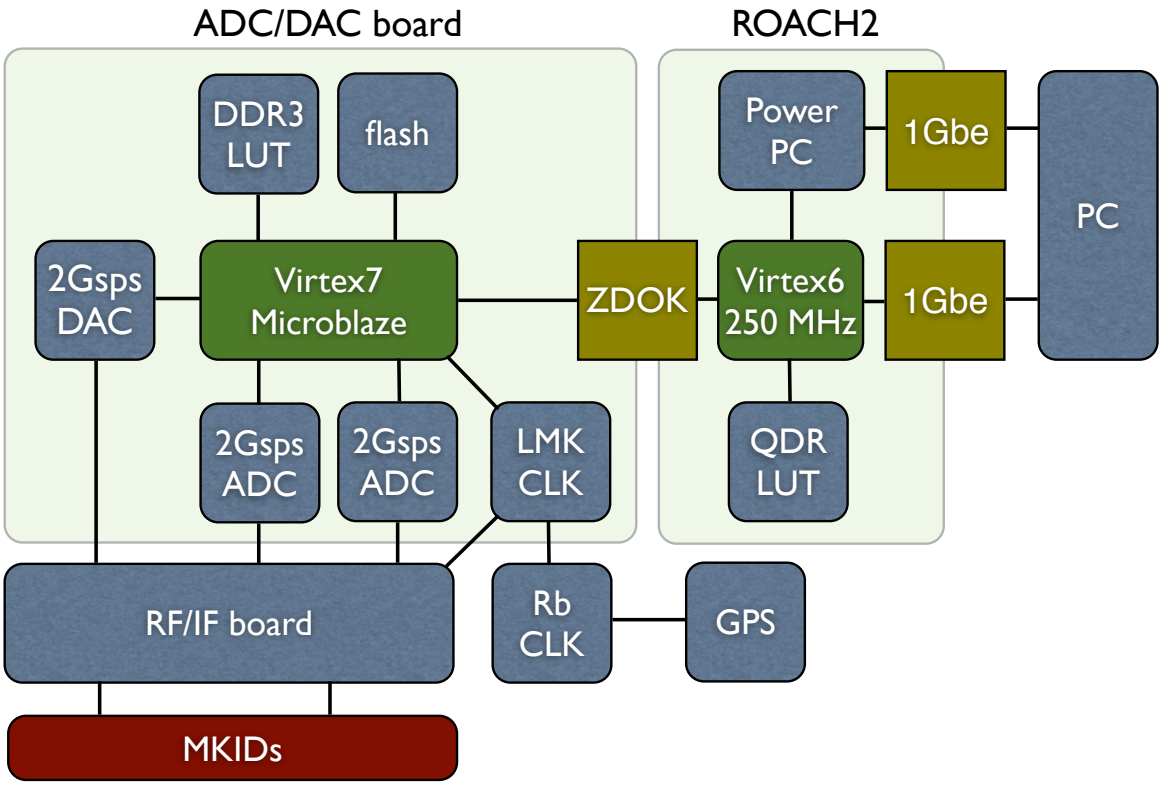

Figure 3.1: Three circuit boards are used. The ROACH2 board houses a Virtex-6 for processing signals. It connects to a Virtex-7 on the ADC/DAC board via a Z-DOK connector.

obtain a phase sample for each channel every µs or a similarly well defined period. If a photon is detected at a particular phase sample we can be sure of when it was detected and the time resolution with which it was detected. Similarly, on the ADC/DAC board, the DAC must be supplied with DAC samples at a very precise 2 GHz. FPGAs do this naturally, where other options would have to be adapted to the task (See Figure 3.1).

3.2 ROACH2 Board

The ROACH and ROACH2 boards were designed by the CASPER collaboration for real time processing in radio astronomy instruments. In particular the ROACH2 specification and design were motivated by the needs of the Square Kilometer Array (SKA) and

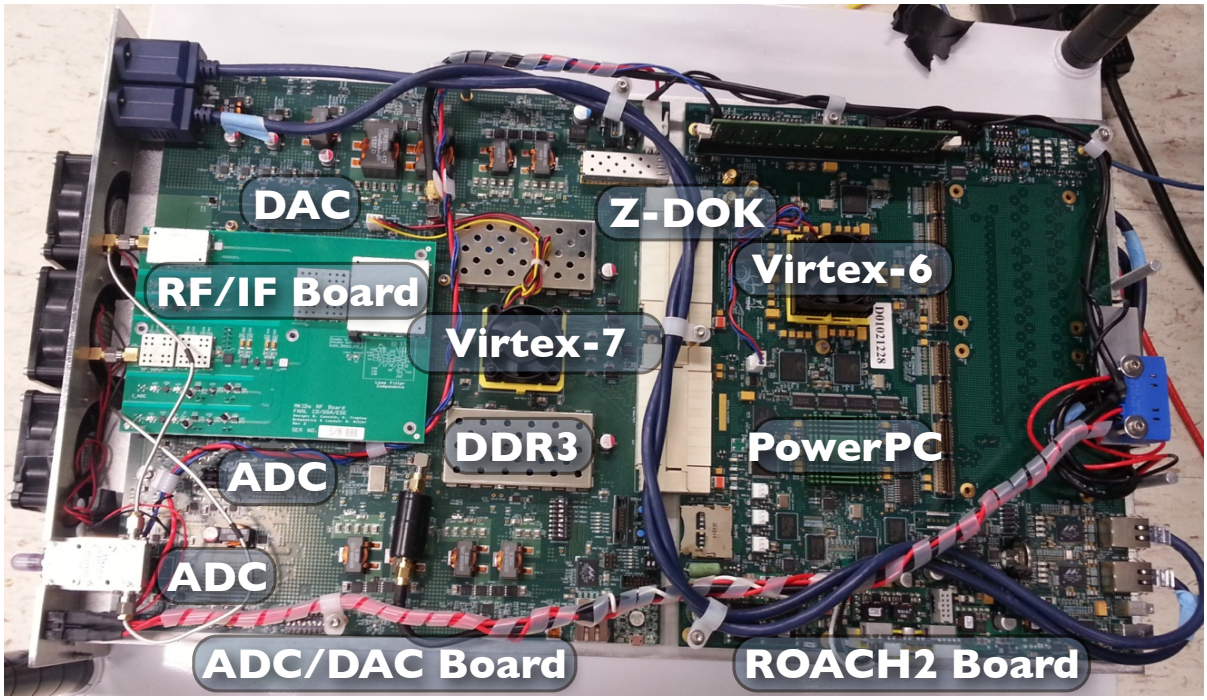

Figure 3.2: The ROACH2 board is connected to the ADC/DAC board by two Z-DOK connectors. The RF/IF board is mounted on the ADC/DAC board using SMP blind-mate connectors for signals and GPIO pins for programming. Another set of three boards are mounted to the underside of this cartridge.

its pathfinders. Each board is equipped with an FPGA, a PowerPC processor, various memory chips, Z-DOK connectors, and utilities for communication such as ethernet. For the ROACH the FPGA is a Xilinx Virtex-5 XC5VSX95T-1FF1136. For the ROACH2 the FPGA is a Virtex-6 XC6VSX475T-1FFG1759C, which has about five times the resources available on the Roach Virtex-5. This FPGA is where most of the processing is done. Its firmware is discussed in Ch 4. Communication and configuration of the FPGA is mainly done through the PowerPC. The ROACH2 is equipped with four QDR II+ SRAM (quad data rate) memory chips directly connected to the Virtex-6. There is also a DDR3 dim chip for the PowerPC.

3.3 ADC/DAC Board

ARCONS made use of ADC/DAC boards designed for the MUSIC Submm MKID project (Duan, 2015). It houses two 16 bit 1 Gsps DACs (DAC5681) and two 12 bit 512 MHz ADCs (ADS54RF63IPFP). In the frequency comb signal generated by the DAC and digitized by the ADC, a single tone may constitute a small fraction of the total DAC/ADC dynamic range. The 12 bit ADC resolution allows for small signals in the comb to be sampled without excessive digitization. The ADC sample rate is what limits the bandwidth readout by a set of boards. For a given spacing of resonators the sample rate limits how many pixels can be read out per readout unit and therefore the cost of the readout per pixel. To move towards large array sizes without exorbitantly priced readouts, we must read out wider bandwidths in each board. Fortunately, over time manufacturers have been producing faster and faster ADCs and DACs for the telecommunication industry. MKID readouts benefit from this continual progress.

For the second generation of MKID instruments, Fermilab has developed a new ADC/DAC board with faster components. It houses a 16 bit 2 Gsps dual channel DAC (AD9136) and two 12 bit 2 Gsps ADCs (AD9625). The two channels of the DAC are used to produced complex I and Q signals. One ADC is used to digitize the I signal and the other digitizes the Q signal. It connects to a ROACH2 with two Z-DOK connectors. A Z-DOK consists of 40 LVDS signal pairs, each of which is capable of transmitting data at 1.25 Gbps. At 2 Gsps with 24 bits per complex IQ sample (12 bits for I, 12 bits for Q), the ADCs generate data at 48 Gbps that needs to be transmitted to the ROACH2. Simultaneously the DAC needs to be supplied a waveform to produce at 64 Gbps. Combined the data rates are higher than the two Z-DOKs can support. This is why a decision was made to place a Xilinx Virtex-7 XC7VX330T-2FFG1761C FPGA on the ADC/DAC board. The purpose of the Virtex-7 is to route the high speed signals

from the ADCs through the Z-DOK to the ROACH2 Virtex-6, and to feed the DACs with values. A Virtex-7 was chosen instead of a smaller FPGA so that it would have enough pins for all signals it would have to route. The Virtex-7 is a generation more advanced than the Virtex-6 on the ROACH2 but for the parts chosen on our two boards the Virtex-6 has more resources. The Virtex-6 has 476,160 logic cells as compared to 326,400 on the Virtex-7.

The ADC/DAC board also has a programmable LMK04821 chip to generate the 2 GHz and 500 MHz clocks for the ADCs and DACs and to serve as the basis for the Virtex-7 and Virtex-6 clock. The Virtex-7 forwards a 125 MHz clock derived from this to the Virtex-6 over a Z-DOK pair, which the Virtex-6 then turns into a 250 MHz clock to be used as it's main fabric clock. The ADC/DAC board takes in a 10 MHz reference signal from a Rubidium clock (See Section 3.5.1). This reference is used to keep the ADC, DAC, and FPGA clocks on multiple boards synced. The ADCs and DAC communicate with the Virtex-7 with a new serial standard JESDb. The outputs of the DAC and inputs of the ADCs also have anti-aliasing filters LFCN-800. These filters remove signals outside of the 2 GHz bandwidth the ADCs can sample so that higher frequencies do not get aliased to lower frequencies and interfere with our intended readout tones.

3.4 RF/IF Board

Fermilab also designed new RF/IF boards to go with the ADC/DAC boards. They mount on top of the ADC/DAC board using SMC blind mate connectors. The resonant frequencies of the MKID array lie in the range 4 to 8 GHz. The ADCs and DACs run at 2 GHz, and so can only digitize signals in the range -1 to +1 GHz, according to the Nyquist/Shannon limit. The RF/IF board houses mixers to mix the DAC output signals up to the resonant frequency range and mix down the returning signals to the

baseband frequency range that the ADCs can handle. Another way to say this is that the board mixes intermediate frequency (IF) signals up to radio frequency signals (RF) and vice versa. The signals are mixed using a local oscillator (LO) frequency generated by TRF3765 chip. This chip also receives the stable 10 MHz reference via the ADC/DAC board. The TRF3765 can generate frequencies up to 4.8 GHz, so a frequency doubler HMC158C8 is used to extend its range to 9.6 GHz.

The mixers are Hittite HMC525LC4 chips, chosen because they work in the frequency range 4 to 8.5 GHz. This range is what limits how many readout units can be connected to a single feedline. For DARKNESS two readout units are connected to each feedline using power combiners and power splitters. Each ADC/DAC can handle a 2 GHz bandwidth, so one unit is used to readout resonators in the 4 to 6 GHz range, and the other in the range 6.2 to 8.2 GHz.

The RF/IF board also has programmable attenuators PE43705 on its RF input and output. These are used to fine tune the tone powers sent to the MKIDs and the tone powers taken in by the ADCs. Each attenuator can be set to an attenuation between 0 and 31.75 dB. The RF output has two variable attenuators in series (for total attenuation up to 63.5 dB) and the RF input has one.

The RF/IF board also has a sequence of fixed attenuators and amplifiers for the RF input and output. The attenuators are included to prevent standing waves from arising and adding noise to the system. The amplifiers adjust for these attenuations and ensure that the right amount of power arrives as each component.

The optimal power for a tone used for reading out an MKID is just below the MKID's saturation point. Past the saturation point the MKID's inductance becomes nonlinear. The resonator is then bistable and the phase tends to snap back and forth between two fixed positions. The saturation power depends on the quality factor, and for our usual devices it ranges from $-100\,\mathrm{dBm}$ to $-90\,\mathrm{dBm}$ at the device. We usually have about

30 dB of attenuation in the cryostat, so the power of single tones going into the cryostat should be in the range −70 dBm to −60 dBm. After the MKIDs a HEMT (high electron mobility transistor) amplifier amplifies the signal by 35 dB. The ADC chips maximum input voltage range is 1 V peak-to-peak (V_{pp}) which for a 50 Ω impedance equates to 4 dBm. The RF/IF board provides the amplification necessary to reach this to use the full dynamic range of the ADC.

3.5 Miscellaneous Hardware

3.5.1 Timing Hardware

A Spectracom SecureSync system [1] provides a stable 10 MHz reference signal to the readout. This is used to generate clocks for the readout boards and to keep everything synced together. The SecureSync also attaches to an external GPS (global positioning system) antenna, to receive an accurate time. The SecureSync has a PPS (pulse per second) output. This is a 5 V TTL (transistor-transistor logic) square wave with a 20% duty cycle, of which the positive edge aligns to the beginning of a new second. This is given to the ADC/DAC boards which pass it along to the Roach2. The PPS and accurate Virtex-6 clock derived from the 10 MHz reference allow the Virtex-6 to time-tag photons with a timestamp accurate to 2 µs (once firmware and filtering delays are removed). The precision is limited by the bandwidth and sample rate of the phase timestream of each channel in the firmware.

[1] http://spectracom.com/products-services/precision-timing/securesync-time-and-frequency-reference-system

3.5.2 Distribution Board

Another custom circuit board distributes power and reference signals to all the readout units. Commercial distribution amps are available, but we wanted one that would fit at the backplane of our readout crate and would minimize the number of wires needed. Each ROACH2 board requires a standard ATX power supply. For each we use a picoPSU-80 [2] to convert a 12 V supply to ATX format. A 12 V supply rated at 800 W, connected to the distribution board, provides the 12 V to all the picoPSU units and the ADC/DAC boards. The ADC/DAC board has a simple 12 V input and provides power to its attached RF/IF board. The two of them together can consume up to 60 W. A Roach2 has a maximum power consumption of 125 W, but typically runs with much lower power. When no firmware is running a Roach2 consumes 36 W.

The distribution board uses buffer op amps to split the 10 MHz and PPS signals to ten ADC/DAC boards without appreciable attenuation. After fabrication it was discovered that the 10 MHz outputs picked up some high frequency noise. Off the shelf low pass filters were used to clean the reference signals before the input to the ADC/DAC boards.

[2] http://www.mini-box.com/picoPSU-80

Figure 3.3: Five cartridges (with two sets of readout boards each) such as in Figure 3.2 slide into this electronics crate. Ethernet ports are provided to connect the ROACH2's to a networking switch. There are also SMA inputs and outputs to connect RF signals to instruments. Two BNC inputs connect a 10 MHz reference and 1 pulse per second (PPS) timing reference to all the boards.

Chapter 4

Firmware and Software

4.1 Introduction

In this chapter, I describe the firmware for both the Virtex-7 on the ADC/DAC board and the Virtex-6 on the ROACH2. The firmware on the Virtex-6 is described in detail with several screenshots to serve as a guide for the future. The last part of this chapter describes the software used to program and control the readout boards. The main purpose of the Virtex-7 is to feed values to the DAC from memory as instructed by the Virtex-6 and to route ADC data to the Virtex-6. The main purpose of the Virtex-6 is to process the ADC data with the channelization algorithm and pulse description described in Ch 2.

4.2 Virtex-7 Firmware

The Virtex-7 firmware is written as a Vivado block diagram. The firmware features a Xilinx MicroBlaze soft core processor. This is programmed with a C program and is used to configure the various chips (ADCs, DAC, clock chip, etc.). The MicroBlaze clock is

set to 100 MHz. The ADC/DAC board is configured to load its firmware and MicroBlaze program from flash memory and start running them when power is turned on.

The firmware reads from a DDR3 chip in a circular fashion using a DMA (direct memory access) and sends the values to be output by the DAC. The DMA is configured for scatter-gather mode, which means that the information needed for all reads (addresses, read lengths) are stored on the DDR3 as well. The connections between IP (intellectual property) blocks are AXI-4 buses. These buses have information signal along with data lines to support bursty data. The data is read from the DDR3 at 250 MHz, 2 x 8 x 16 bits at a time and are serialized and sped up. The data is then formatted to send to the DAC with JESD IP from Xilinx. Since the clock speeds in the FPGA are lower than 2 GHz, multiple 16 bit DAC samples are transferred between IP in a single cycle. When multiple samples are sent together the less significant bits represent samples to be generated earlier in time. In the DDR3, the DAC samples are stored in temporal order for increasing address. I and Q values for a single complex sample are grouped together with I occupying the 16 more significant bits and Q occupying the less significant bits.

The data from the two ADCs is received into an eight lane JESDb IP. The data forwarded by the JESDb block is sliced and reorganized so that it will come through in the correct order after transferring to the ROACH2. The data is placed in 8:1 OSERDES to be sped up and serialized from 125 MHz single data rate (SDR) to 500 MHz double data rate (DDR; samples on both positive and negative edges). The I and Q from each ADC are divided into two data buses each for a total of four buses. Each bus contains twelve data bits, one bit to indicate an overrange condition, and one bit to forward the sysref signal for synchronization. These four 14-bit buses are connected to FPGA pins leading to Z-DOK pairs.

Another Z-DOK pair is used to forward the 125 MHz clock that was used for the ADC data OSERDESs. Two pairs are used for UART (universal asynchronous re-

ceiver/transmitter) communication between the MicroBlaze processor and the Virtex-6, and one more pair is used as an input to the Virtex-7 so that the Virtex-6 may send a signal that it is ready to receive ADC data. Also, the PPS signal that the Virtex-7 receives from the distribution board is forworded on a Z-DOK pair to the Virtex-6 without change.

The IP blocks to be controlled by the MicroBlaze are connected as peripherals using AXI-Interconnect blocks. Different Interconnect blocks are used to separate peripherals that need to transfer data at different clock rates. Most blocks (e.g. blocks for programming ADCs, DAC, RF/IF board) run from the same 100 MHz clock as the MicroBlaze. The DMA and MIG (memory interface generator) used to quickly move data from the DDR3 chips are connected to a 200 MHz Interconnect.

The pulse called a sysref signal is sent to the ADCs and DAC to synchronize them. If the ADCs were not well synchronized there would be a time offset between the I and Q values digitized. For a pure complex tone with a positive frequency the I sinusoid would lead Q by 90°. If the frequency is negative, Q would lead I by 90°. A time delay would mean that this phase relationship would not hold. Unless the phase difference is exactly ±90°, the complex waveform would be wrongly interpreted as a supperposition of both positive and negative frequencies.

4.3 Virtex-6 Firmware

The Virtex-6 firmware is written in Simulink, a visual programming plugin for MATLAB. The firmware is composed of simulink blocks, which at the lowest level correspond to Xilinx IP modules or files written in a hardware description language (HDL), either Verilog or VHDL. CASPER provides MATLAB scripts to compile simulink model files into bitsream files that can be loaded onto the Virtex-6. They make use of the Xilinx

System Generator.

4.3.1 Yellow Blocks

Inputs and outputs to the Simulink model are handled by special yellow blocks. These contain Xilinx gateway blocks to connect lines to either FPGA pins or other signals in the HDL for the project. The blocks are connected to MATLAB code that intantiates a class object to keep track of what it is connected to and other neccessary attributes such as what timing constraints should be applied to each connection. CASPER provides yellow blocks to connect the Simulink model file to registers and BRAM (block memory inside the FPGA) accessible via memory mapping by the PowerPC, which then can be controlled remotely with CASPER's KATCP commands.

Two yellow blocks were developed for the ROACH2 to communicate with the ADC/DAC board. The first brings the ADC data into the Simulink block. The second implements a UART to send instructions and data from the Virtex-6 to the Virtex-7. A MATLAB script populates the necessary class object with a list naming particular Virtex-6 pins as inputs and outputs of a Verilog module and a list of the gateway inputs and outputs in the Simulink block, which are also connected to the Verilog module. There is a MATLAB matrix included in the base ROACH2 package that matches Z-DOK pairs to Virtex-6 pin names. Indices to this matrix are used to indicate which Z-DOK pairs are labelled as particular bits in the four 14-bit ADC data buses, or one of the other control signals. In the `adcdac_2g` module, a Mixed Mode Clock Manager (MMCM) takes in the 125 MHz clock provided by the ADC/DAC board via a particular Z-DOK pair, and uses it to create six new clocks that are matched in phase. One is the same frequency, 125 MHz. Another is four times as fast at 500 MHz. The other four are all at 250 MHz at different phases, 0°, 90°, 180°, 270°. These last four are used as the FPGA fabric clock. The 500 MHz

and 250 MHz (0° phase) clocks are used in 4:1 ISERDES modules to parallelize and slow down the four data buses from coming in at 500 MHz DDR (samples on both positive and negative clock edges) to 250 MHz SDR (single data rate; samples only on positive edges) to match the fabric clock. This way, eight 12-bit ADC samples (plus 2 status bits) from both I (buses 0 and 1) and Q (buses 2 and 3) enter the Simulink model each cycle. One of the parameters to the MMCM instance in Verilog is intentionally set incorrectly. The MMCM parameters determine the factors to divide and multiply the input frequency to produce all the necessary output frequencies. Only certain combinations of factors may be used. The MMCM first divides the input frequency to produce f_{pfd} (phase frequency detector) before multiplying by another set of factors to make the desired outputs. The MMCM paramters are constrained such $f_{\text{pfd}} \geq 135\,\text{MHz}$. Otherwise, compilation errors out. Xilinx added this constraint because certain Virtex-6 chips would not behave properly for lower f_{pfd}. However, to produce the frequencies mentioned above, this f_{pfd} must be 125 MHz. So, in the Verilog parameters, it is set to 135 MHz, but it is actually given a 125 MHz input. This seems to work well for our chips.

All of the inputs from Z-DOK pairs must each first pass through an IBUFDS to convert them from differential signals to simple logic signals. At the ISERDES modules, the data in the four data buses are not necessarily aligned perfectly with the 500 MHz clock edges. Each bit must travel a slightly different trace length. In fact the trace lengths for Z-DOK 0 are about 1.8 inches longer than for Z-DOK 1. This results in offsets on the order of hundreds of picoseconds in the edges of each bit. The ISERDES samples each data bit on the positive and negative edges of the 500 MHz clock. If two bits flip at the same time at the ADC but gain offsets relative to each other and the clock before reaching the OSERDES, they may be seen to flip in separate clock cycles. This means a 12-bit ADC sample is not correctly reconstructed for at least one cycle as it passes into the Simulink model. To prevent this, every data bit is passed through an

IODELAY module before entering the ISERDES. The IODELAY has a programmable delay with 32 settings, from 0 ps to 2500 ps delay.

The same set of IODELAY settings may be used for all test ROACH2 and ADC/DAC combinations with one caveat. The data coming over the Z-DOK is slowed from 56 bits (two I/Q pairs plus overrange and sysref bits) at 500 MHz (DDR) to 224 bits at 125 MHz (SDR). The slower clock is phase locked such the positive edge of one matches the positive edge of another. However, there are two possible edges for the slower clock to match to. If it locks to the wrong one, the 56 bit transfers will be packed into 224 bit words in the wrong order. We fix this by dynamically changing the MMCM phase, so that if it locks to the wrong edge it can be shifted by one 500 MHz edge to the other edge. Two inputs are included in the yellow block for dynamically shifting the MMCM phase.

4.3.2 Simulink Digital Signal Processing

Along with yellow blocks that interface with hardware via Verilog modules, the Simulink design contains white blocks, blue blocks, and green blocks (See Figure 4.1; Later figures show grayscale blocks). White blocks are subsytems that contain more simulink blocks, and are used simply to modularize the logic. Blue blocks are provided by Xilinx and perform simple operations such as bit manipulation, arithmetic, and logic. Green blocks are given parameters when double-clicked and are connected to Matlab scripts that rewrite the block depending on the paramters. Most are provided by CASPER libraries, though a few are custom made (e.g. the programmable FIR block with a parameterized number of taps). Several screenshots of the Simulink design follow to illustrate the implementation of the principles desribed in Chapter 2. A zoomed out view of the toplevel of the Simulink design is shown in Figure 4.2.

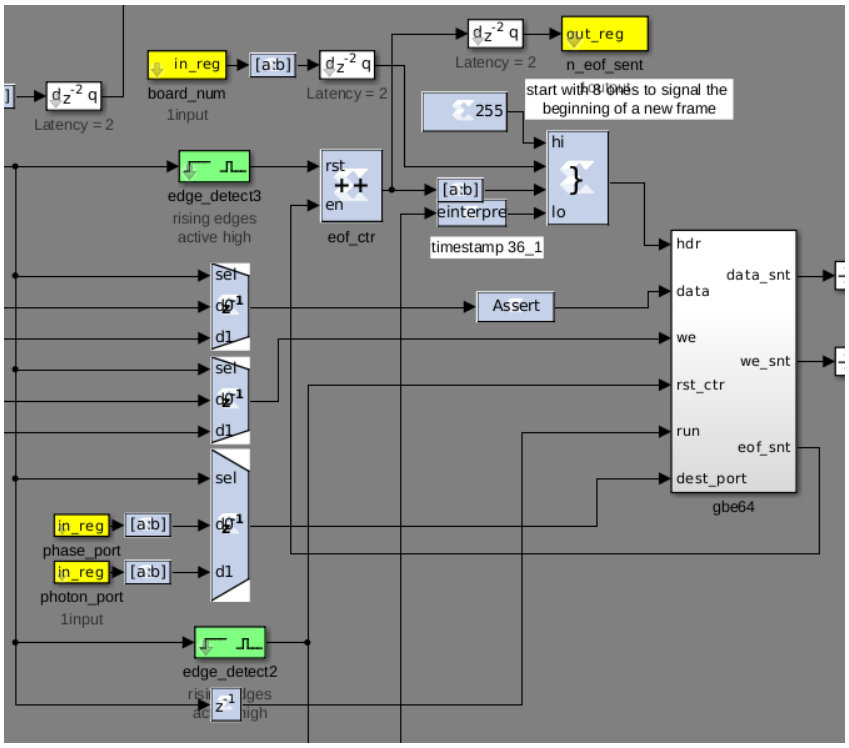

Figure 4.1: A screenshot displaying Simulink blocks of various colors. The edge blocks change their internal operation depending on given parameters. The blue blocks are Xilinx blocks. The yellow blocks shown are registers that can be set or read by the DAQ computer. The white block encapsulates other logic.

UART Control of the ADC/DAC board

The logic and yellow block for sending UART controls to the Virtex-7 is separated from the rest of the Simulink design and runs independently. The logic is placed inside the `a2g_ctrl` subsystem in Figure 4.4 and is shown in Figure 4.3. Eight bits can be given to the UART yellow block in a particular cycle. This cannot be done often, though, because the baud rate is 921600. Because each set of eight bits is signaled with one start bit and one stop bit, this baud rate corresponds to 92160 bytes transmitted per second. The logic in the `a2g_ctrl` block slows transfers to match this rate. Either single bytes or a pre-written LUT can be sent. Sending single bytes is used to send commands to the MicroBlaze on the Virtex-7. Each command is one byte. Some commands are followed

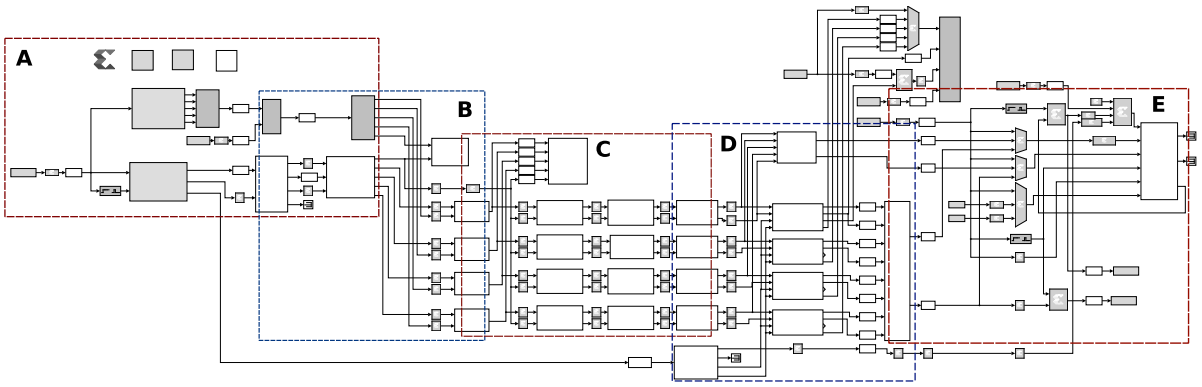

Figure 4.2: A zoomed out view of the toplevel of the Virtex-6 Simulink design. Red and blue rectangles indicate sections shown in later figures.

by data. For example if the command is to change the variable attenuation on the RF/IF board input, the new attenuation follows the command. If the command is to load the DAC LUT, the entire DAC LUT follows the command.

At the moment, the length of the DAC LUT is hard-coded in the MicroBlaze code to be 2^{18} I/Q pairs. I and Q are 16 bits each, so the whole LUT requires 512 kB. This requires multiple turns of writing to the BRAM in `lut_dump`, and sending it to the UART. The length of the DAC LUT sets our DAC frequency resolution, because we do not want any sudden discontinuities when the DAC LUT finishes playing out the DAC and starts over from the beginning. To make a smooth transition at the boundaries, the periods of all tones in the LUT must fit exactly in the LUT. So, all periods must obey

$$\frac{T}{dt} = \frac{N}{m}$$

for some integer m, where T is the tone period, $N = 2^{18}$ is the number of samples, and $dt = 0.5\,\text{ns}$ is the time between DAC samples. This can be rewritten for the tone

frequency $f_{\text{tone}} = 1/T$ and sample rate $f_s = 1/dt$:

$$\begin{aligned} f_{\text{tone}} &= m \, \frac{f_s}{N} \\ &= m \, \frac{(2\,\text{GHz})}{2^{18}} \\ &= m \, (7.63\,\text{kHz}). \end{aligned}$$

So, the frequency resolution of the DAC is 7.63 kHz. We will require the DDS frequency resolution to be the same.

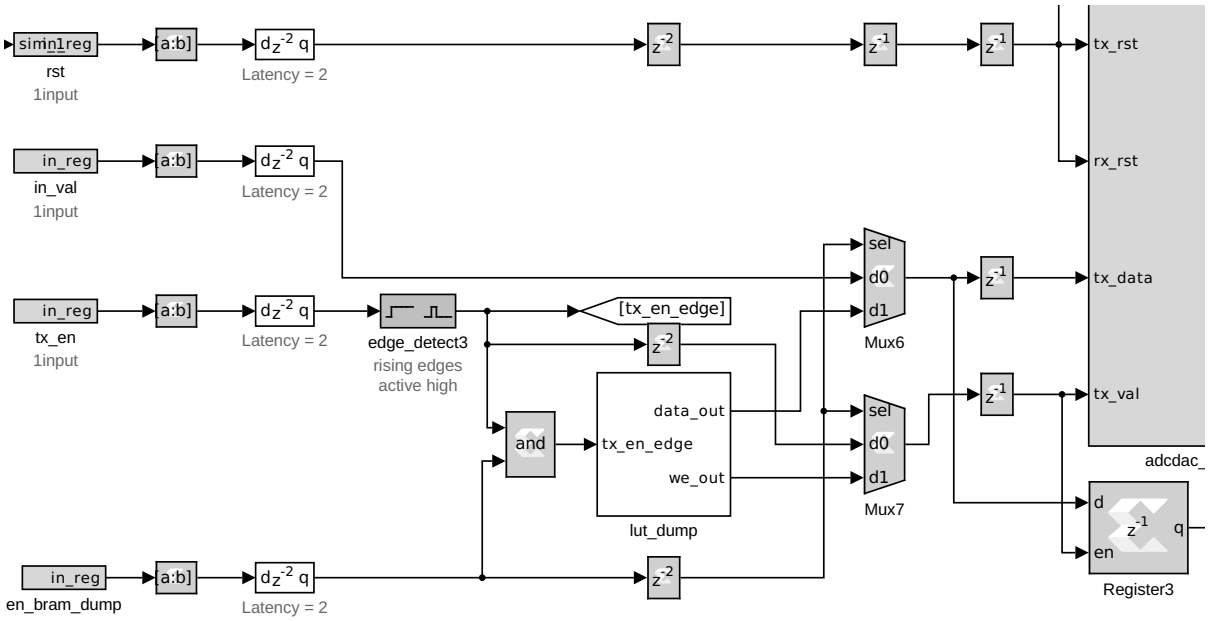

Figure 4.3: The UART ADC/DAC yellow block (large block at right edge) can send one byte to the ADC/DAC board. The logic shown can send a single byte or send a pre-written look up table one byte at a time.

Receiving Data from the ADC

Figure 4.4 shows the part of the toplevel design where data is received from the ADC and sent into the PFB. The interior of `adc_in` is shown in Figure 4.5. The firmware runs at 250 MHz. Eight I/Q pairs are received in the block each cycle. Each I and Q is 12

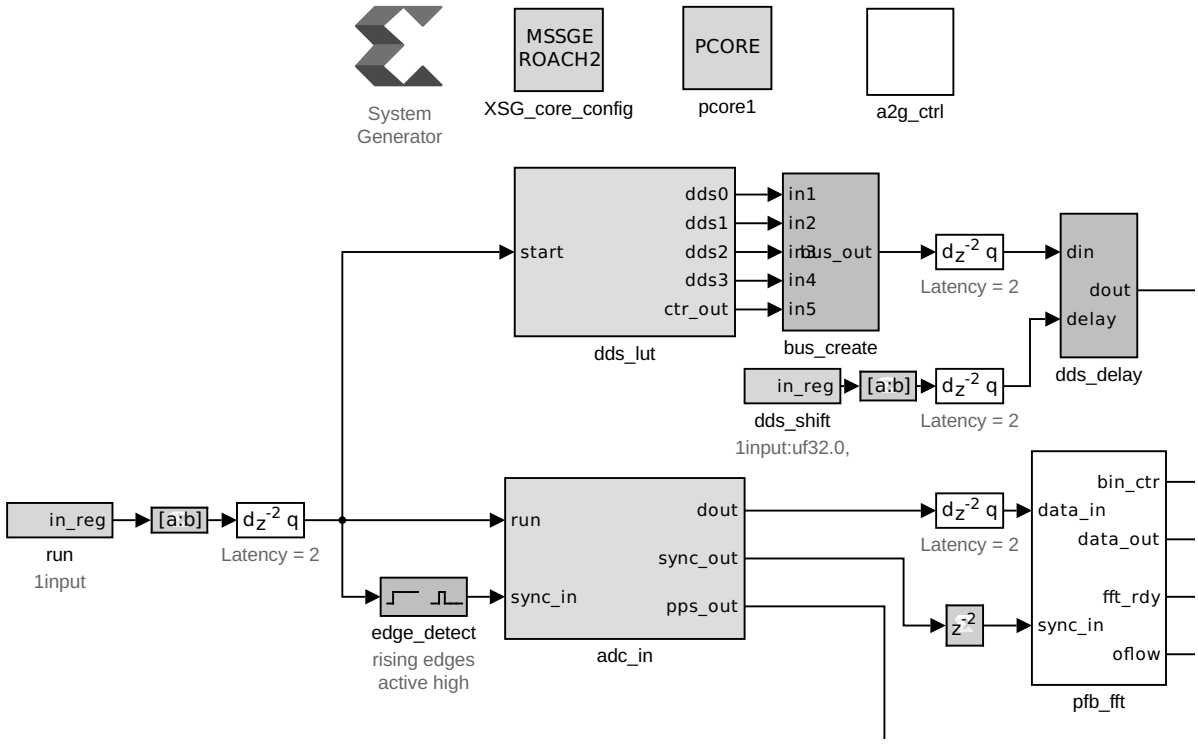

Figure 4.4: Selecton A from Figure 4.2. Data from the ADC is sent in a bus to the PFB to perform the coarse channelization. In the `dds_lut` block, the DDS look up table for all channels is read and then delayed by `dds_delay`.

bits. The overrange and sysref bits sent with each I and Q are ignored. The eight pairs are concatenated into a single bus of width 192 bits to send to the `pfb_fft` block. In the area not shown below the logic in Figure 4.5 there are snapshot blocks to capture a small series (8192 samples) of raw I/Q values from the ADC block. This is useful for debugging.

There is an option to scale the I/Q values received from the ADCs by a factor less than one. This was added after we found that the FFT was very sensitive to overflowing when data containing pure tones was given as input. The effective number of bits (ENOB) for the ADCs is 9.2 bits (at 500 MHz) out of 12 bits, so we do not lose any information by shifting down a couple of bits at this stage. Scaling any lower will increase quantization noise.

One output (labeled `sync`) from the ADC yellow block provides the PPS signal from the Spectracom SecureSync propagated through the ADC/DAC board. At this point it still has a long duty cycle. In the firmware, we want a PPS signal consisting of a single cycle pulse to indicate the beginning of a new second. An edge detect block turns the given PPS signal into the desired pulse. Logic is also included to check if somehow a pulse has been missed. If it does not receive a pulse in more than a second, it assumes it's been missed and creates a pulse itself. For the purpose of absolute timing, this is not ideal, but for most observations counting FPGA clock cycles derived from the 10 MHz reference will be more than accurate enough to tell when new seconds begin. There is also a register here to indicate if PPSs from the ADC have been missed and inserted.

Coarse Channelization

Figure 4.6 shows a Xilinx black box block that performs the PFB. The two FFTs with two preceding PFB FIRs are in a separate Simulink design. That design was synthesized into a netlist, and the netlist is included in this design. This saves time when compiling the full firmware, since the PFB takes a long time to synthesize. As mentioned in Section 2.2, a four tap PFB FIR is used with a Hamming window, and the bin width is widened by a factor of two. Then two 2048 point FFTs are performed with the input data of one offset from the other by 1024 samples. Two complex samples from eight bins are output per clock cycle per FFT. The 2048 bins are in canonical order, starting from the smallest postive frequency and increasing to the maximum. Then the second half of the bins contain the negative frequencies starting with the most negative. The sample rate

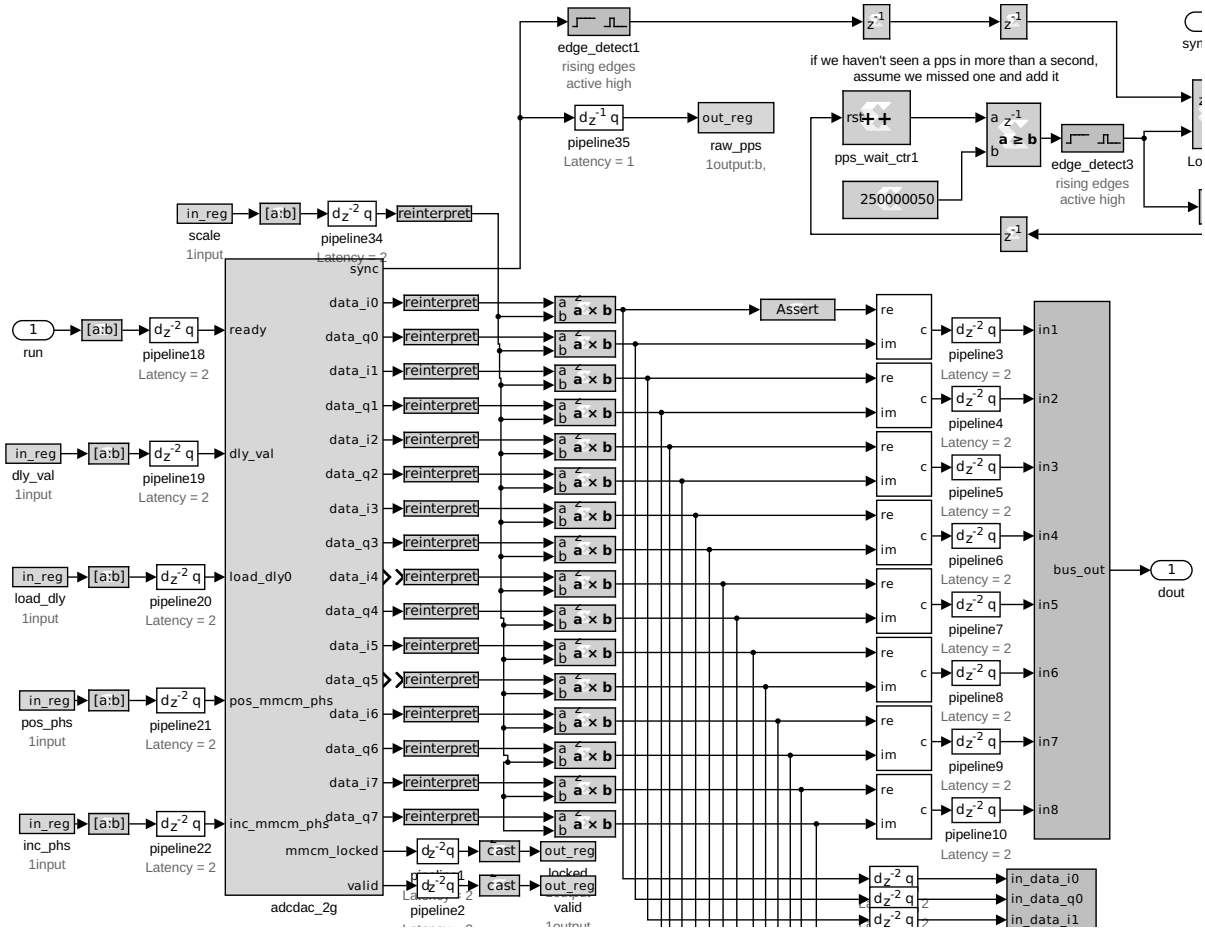

Figure 4.5: The interior of the `adc_in` subsystem in Figure 4.4. The data ADC/DAC yellow block (large block on the left) outputs eight I/Q pairs sent from the ADC/DAC board through the Z-DOK. These are scaled (blocks labeled $a \times b$) and packed into a bus (tall block on the right). PPS logic is at the top.

for a particular bin is given by

$$f_{\text{s,bin}} = \frac{(2\text{ FFTs})(1\text{ bin sample per FFT})(250 \times 10^6\text{ cycles per s})}{2048/8\text{ cycles to loop back to the same bin}}$$

$$\approx 2 \times 10^6\text{ bin samples per s.}$$

This gives the needed 2 MHz sample rate we established in Section 2.2.

Each bin sample has an 18 bit real component and an 18 bit imaginary component. I will continue to use the I/Q terminology for these complex values to maintain consistency

with the earlier treatment of complex values. With two values per bin and eight bins per cycle, the data bus coming out of the `pfb_fft` block is 576 bits wide.

A register is used to indicate if either of the FFTs overflowed since the register was last reset. Bins with coherent tones grow in each stage of the FFT, which makes overflows more likely. The CASPER FFT can be set to shift down intermediate results after each stage to prevent overflow. The FFT is set to dynamically determine its shift schedule, so it will try to decide when it needs to shift down. There is also a shift input pin that affects how often it is shifted down. At the moment the input to this shift pin is hardcoded. The usual value we use (31, corresponding to 7 down shifts) probably is not enough to prevent all overflow for our usual tones. At the other extreme, the maximum possible value shifts too many times causing the FFT output to scale to near zero. More testing with a variable shift pin should be done to determine the optimum value when 1024 coherent tones are put in. In the short term, scaling the ADC signal down a few bits also helps to prevent overflow.

Channel Selection

Figure 4.7 shows the 576 bit data bus containing FFT bin values continuing to the channel selection step. Of the 2048 FFT bins, 1024 are selected as channels, depending on which bins contain readout tones. Some FFT bins may contain multple tones, and so will be selected and copied into multiple channels (See Figure 2.2). The `chan_sel` block contains a bit of BRAM that is configured to keep track of which bins to keep and in what order. At this point we not only choose 1024 bins to become channels, but separate these channels into four lists of 256 channels each. After this block each list will be processed separately and independently. The 256 channels in each list are time multiplexed, so after this it takes only 256 cycles to loop through all channels. This splitting into four lists is necessary to make full use of the parallel resources of the FPGA and to keep the

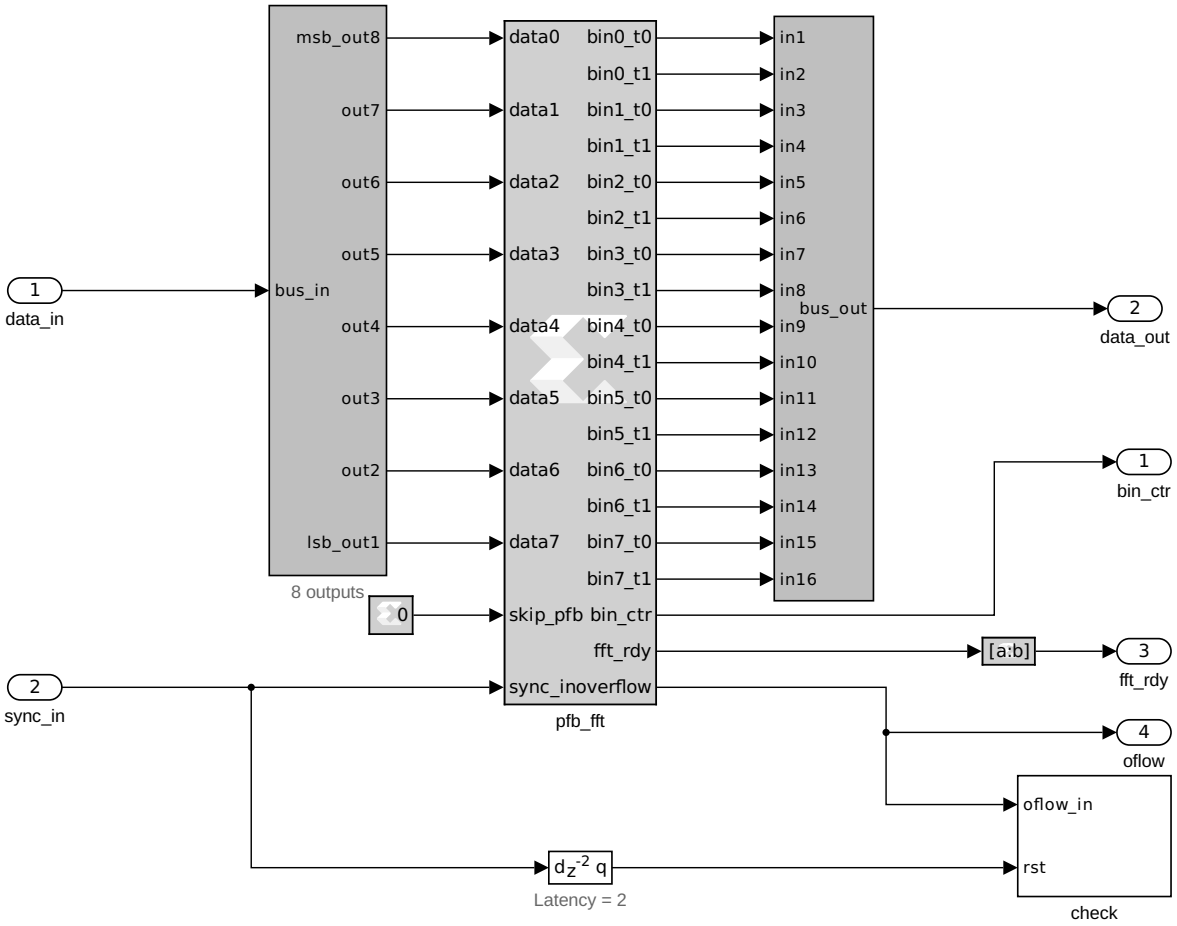

Figure 4.6: The inside of the `pfb_fft` block in Figure 4.4. The two PFB FIRs and FFTs are compiled into netlists separately, and then included here as a black box.

final phase sampling rate as high as it was in ARCONS. If we only had one list that took 1024 cycles to loop through, the final sampling rate would be four times slower.

The `chan_sel` block contains three layers of buffer subsystem blocks to store and retrieve the needed bins. In the highest layer it contains two buffer blocks. In any given cycle one is being written to, and the other is read from. This way there is no interruption of data flow while reading or writing. In each of these read and write buffer blocks, there are four buffers. Each of the four handles a different list of 256 channels. All of the bins must be written to all of them, though, so that we can be maximally flexible in assigning any bin into any of the lists of 256 channels. In each of the four list buffer blocks, we

arrive at the bottom level, which contain eight blocks of memory to match the eight bins that come from the `pfb_fft` block in a cycle. Two I/Q pairs are written to each of these eight blocks when in writing mode. When reading, only one of the eight is read out at a time depending on which bin is desired to become the current channel.

At the end of this we now have four buses handling the four lists of 256 channels. In a given cycle one of these buses will contain two 36 bit I/Q pairs from a single channel. Our 2 MHz sample rate is maintained.

Mixing with the DDS

Figure 4.8 shows how the DDS LUT is read from four QDR (quad data rate) chips. There are exactly four QDR chips on the ROACH2 and we use them all for this purpose to achieve the needed bitwidth per cycle. A single QDR Simulink block can effectively read or write 64 bits per cycle. Each could in fact read 72 bits per cycle in firmware, but only 64 of these bits can be written using the CASPER KATCP libraries, so we only use these 64 bits. Each I and Q DDS value is 16 bits, so one QDR can output two I/Q pairs per cycle. The values are time multiplexed by channel. There are four sets of 256 channels each. Each QDR is assigned one of the four lists of channels. To match the data coming out of the `chan_sel` block, the two I/Q pairs from a single QDR are two samples for a single channel.

One extra complication is that the QDR reads and writes in bursts of two cycles. Only one address is presented per burst. Extra logic is present in Figure 4.8 to ensure that a burst is not interrupted. We store 2^{20} 64-bit words in each QDR (which corresponds to 2^{19} address). It takes 256 cycles to loop through to the same channel. As in the DAC look up table, the frequency resolution is determined by what frequencies will fit

perfectly in the LUT. The time between reads on a particular channel is

$$\begin{aligned}\Delta t &= \frac{256 cycles}{f_s} \\ &= \frac{256 \text{ cycles}}{250 \times 10^6 \text{ cycles per s}} \\ &\approx 1\,\mu\text{s}.\end{aligned}$$

Since we read two samples for a bin each cycle the sample rate is $f_{\text{s,dds}} = \frac{2 \text{ samples}}{\Delta T} = 2\,\text{MHz}$. The number of samples for a particular channel in the LUT is

$$\begin{aligned}N &= \frac{(2 \text{ samples per QDR word})(2^{20} \text{ QDR words})}{(256 \text{ channels in a QDR})} \\ N &= 8192.\end{aligned}$$

Now we can calculate the DDS frequency resolution:

$$\begin{aligned}\Delta f_{\text{dds}} &= \frac{f_{\text{s,dds}}}{N} \\ &= 238\,\text{Hz}.\end{aligned}$$

We need the DDS frequencies to be able to take on any values of the bin may oscillate at, $f_{\text{bin,osc}} = f_{\text{tone}} - f_{\text{bin center}}$. Both f_{tone} and $f_{\text{bin center}}$ are both integer multples of Δf_{dds}, so the resolution is good enough. In fact we could reduce the DDS LUT to 2^{15} QDR words so that the DDS frequency resolution matches the DAC resolution 7.63 kHz.

The DDS signal is delayed by a variable delay block before reaching the mixers (See the `dds_delay` block in Figure 4.4). The delay takes on a value between 0 and 512 cycles. The purpose is to mix the 256 interleave signals in each DDS bus with the correct FFT bins (also interleaved) from the `chan_sel` block. In other words, a DDS sample from channel 0 should arrive at the mixer blocks in the same cycle as an FFT bin sample from

channel 0. If this dds_delay is not set to the right value and, for example, a DDS signal from channel 0 gets mixed with the FFT bin data from channel 23, then it is likely that the resulting signal will not be 0 Hz as desired. In all likelihood the resulting signal will be killed by the later low pass filter. The correct delay is found empirically, and set by a yellow block register. When the delay is wrong, we can usually tell, because the I/Q loops we acquire from `acc_iq` (discussed below) all turn out to be small noisy bundles of points. Usually a few loops still look like loops just by chance.

At the right edge of 4.7, the selected channels, separated into groups of 256, are multiplied with the DDS signal from the QDR. The output is found as

$$I_{\text{out}} = I_{\text{dds}} I_{\text{bin}} + Q_{\text{dds}} Q_{\text{bin}}$$
$$Q_{\text{out}} = I_{\text{dds}} Q_{\text{bin}} - I_{\text{bin}} I_{\text{dds}}$$

Given the definitions

$$(I_{\text{bin}} + iQ_{\text{bin}}) = e^{2\pi i (f_{\text{tone}} - f_{\text{bin center}})t + i\phi_{\text{bin}}}$$
$$(I_{\text{dds}} + iQ_{\text{dds}}) = e^{2\pi i (f_{\text{tone}} - f_{\text{bin center}})t + i\phi_{\text{dds}}}$$

the mixers perform the complex multiplication

$$\begin{aligned}(I_{\text{out}} + iQ_{\text{out}}) &= (I_{\text{bin}} + iQ_{\text{bin}})(I_{\text{dds}} - iQ_{\text{dds}}) \\ &= e^{2\pi i ((f_{\text{tone}} - f_{\text{bin center}})t + i\phi_{\text{bin}}} e^{-2\pi i (f_{\text{tone}} - f_{\text{bin center}})t - i\phi_{\text{dds}}} \\ &= e^{i(\phi_{\text{bin}} - \phi_{\text{dds}})}.\end{aligned}$$

Constant scale factors have been ignored. The `mixer` blocks perform its multiplication

two I/Q pairs per cycle. The effect of mixing is to shift the frequency of interest to 0 Hz. The negative sign inserted at the imaginary part of the DDS signal in the multiplication changes the sign of the DDS frequency for the multiplication. This makes setup a little easier. If the channel signal of interest is oscillating at $f_{\text{bin,osc}} = f_{\text{tone}} - f_{\text{bin center}}$, then we multiply by a DDS with the same frequency with the same sign to shift it to 0 Hz.

We will soon convert to phase, and ϕ_{bin} will be the signal we are interested in. Notice that ϕ_{dds} is a value we can choose (and can be different for every channel) to apply an offset to ϕ_{bin}. We use this when setting up the readout. When defining the DAC LUT we randomize the initial phases of the tones in the comb. We empirically determine the phases of each channel after the `mixer` blocks using the `acc_iq` block (seen in Figure 4.10).

The `acc_iq` contains four vector accumulators. These collect running averages of the last 1024 I and Q values for each channel. These can then be read with snapshot blocks. When setting up the readout, we usually step the LO through a sequence of frequencies (centered on our nominal f_{LO}), and collect an averaged I/Q point at each frequency for each channel. Plotting Q vs I for a sequence of frequencies around the resonant frequency reveals a resonant loop (See 4.9). The loop encodes both the transmission and phase response of the resonator as a function of frequency. Transmission would be the distance of each point to (0,0). We construct loops for all 1024 channels, and from them can find the loop center I/Q and the phase at the resonant frequency (using Eq 2.3). We then set ϕ_{dds} for each channel in the DDS LUT such that the baseline phase $(\phi_{\text{bin}} - \phi_{\text{dds}}) = 0$.

Filtering and Downsampling

Figure 4.10 shows the continued parallel processing of the four lists of 256 channels each. After multiplying with the DDS signal in the `mixer` blocks, the fine channelization

is completed by passing both I and Q through low pass filters and downsampling in the `downsample` blocks. Performing the low pass filter is only safe once the frequency of interest has been shifted to 0 Hz. Any other frequencies from other tones that happened to be in the same FFT bin will be cut out of the channel at this time. (In a different channel that other tone is the frequency of interest shifted to 0 Hz and the current frequency of interest is the nuisance frequency that is cut out.)

This works as long as tones are separated by at least 500 kHz. When two MKID resonant frequencies on the same feeldine happen to be closer than this, we only create a readout tone for one of them, and the other MKID is left as a dead pixel. If somehow there is another tone less than 500 kHz from the frequency of interest, then the low pass filter will not effectively cut it out, and rather than be mixed to a constant value, the I/Q values in the current channel will continue to oscillate. This oscillation will dominate the final phase signal, and photon phase pulses will be impossible to discern. We call a pixel in which this has happened a "beater," because the phase will continuosly wrap around 2π according to the beat frequency $f_{\text{nuisance tone}} - f_{\text{good tone}}$.

The low pass filters are 20 taps, only 18 coefficients of which are nonzero. The coefficients are hard-coded in simulink to implement a low-pass FIR filter with a cutoff frequency $f_c = 250\,\text{kHz}$. This will limit the bandwidth of signals present in each channel. It is then safe to downsample. In the `downsample` blocks, the filtering and downsampling (by 2) is done simultaneously to conserve resources. Two I/Q sample pairs per cycle enter the FIR, but only one filtered I/Q pair emerges each cycle.

The FIRs are designed to work on 256 interleaved (time-multiplexed) signals. An FIR performs the operation

$$I_{\text{filtered}}[n] = \sum_{m=0}^{M-1} h[m] I_{\text{raw}}[n-m].$$

where n is the time (or cycle), and there are M FIR coefficients h. But for the time-multiplexed channels, consequtive samples for a particular channel do not appear in consecutive samples. If the current sample is from channel m, we have to go back 256 cycles to find the last sample from the same channel, and another 256 for the sample before that. So the correct FIR implementation would perform

$$I_{\text{filtered}}[n] = \sum_{m=0}^{M-1} h[m] I_{\text{raw}}[n - 256m].$$

The firmware uses many 256 cycle delays to do this. The delays here and in other parts of the firmware (e.g. the programmable phase FIR) represent a significant portion of the FPGA resource utilization. The default Xilinx delay block in Simulink is implemented with shift registers. Since shift registers are used widely through the firmware, and BRAM is less used by the other portions of the firmware, I usually find it more efficient to implement these 256 cycle delays with BRAM delay blocks.

Pulse Detection

After filtering I and Q and downsampling, we convert to phase accroding to Eq 2.3. The `arctan` operation is done with a Xilinx CORDIC block. The I/Q centers found from using the `acc_iq` block are load and subtracted before the CORDIC block. After these blocks we have four phase timestreams time-multiplexed with 256 channels each. These are then filtered by a programmable filter in `prog_fir` blocks. These blocks contain a scripted block. Changing the number of taps in the block parameters runs a MATLAB script to rewrite the interior components of the block to implement this. This makes it simple to change the size of the FIR. The `prog_fir` blocks allow for a different set of filter coefficients to be applied to each channel. (See Section 2.3.1 for a discussion of the optimal filter coefficients used). The latest firmware is compiled with 50 tap FIRs,

where each coefficent is loaded into BRAM using registers. The many multiplications and delays consume significant FPGA resources and can cause difficulties when making timing (See Section 4.3.3). When I add new features to the firmware, I often decrease the number of taps to ensure I can compile and test the new features. Then I increase the number of taps and try to compile again.

Figure 4.11 shows the time-multiplexed phase timestreams continuing out of the `prog_fir` blocks and into `capture` blocks for pulse detection. The `capture` blocks implement the SVF filter discussed in Section 2.3.1 to find the baseline of each channel's phase. It then checks the trigger conditions listed in Section 2.3.2. The peak trigger conditions are checked in the `trigger` block in Figure 4.12. Phase thresholds are calculated from snapshots of phase from each channel and are loaded into the `capture` block to compare with. Once a peak is detected below the appropriate threshold, the `deadtime` block ensures that there is not another trigger in the same channel for at least 10 µs. The triggers for each channel per second are counted, and the `lim_cps` block sets a maximum counts per second.

The lower portion of Figure 4.12 shows the construction of a 64 bit "photon" word. When all the trigger conditions are met, this 64 bit word is accompanied by a signal `we_out=True`. This word is what will be written to disk to record that a photon has been detected. The structure of this word is very different from what was used in ARCONS (See Figure 3 in van Eyken et al., 2015 for the ARCONS format). In the new format the most significant 20 bits in the word indicate what pixel the photon was detected in. Usually these 20 bits give the spatial location of the pixel in the MKID array (10 bits for the x coordinate and 10 bits for the y coordinate). The coordinates are loaded into BRAM (labelled `pix`) and indexed by the channel number within the firmware. The next 9 bits are a timestamp indicating when the photon was detected. These bits record the number of µs since the last half ms. The next 18 bits are the phase at the

detected pulse peak. It is formatted as `fix18_15` value. This means that is a signed (2's complement) fixed point binary number with 18 bits total, and 15 bits after the binary point. That is 1 sign bit, 2 bits for the integer part, and 15 bits for the fractional part. The least significant 17 bits are phase baseline computed by the SVF filter. This is kept for debugging purposes. The peak phase in the word has already had the baseline subtracted. By keeping the baseline we could add it back and check if there was anything wrong with the baseline for this channel.

The `timekeeper` block in Figure 4.11 generates the 9 bit timestamps for photons and 36 bit timestamps for ethernet frames. Because the FPGA fabric clock is ultimately derived from the LMK04821 chip on the ADC/DAC board, which itself is locked to a stable 10 MHz reference from a Rubidium clock, the fabric clock is reliable for use in timing photons with sufficient accuracy in the short term. The `timekeeper` block counts the number of cycles since the last PPS signal from the SecureSync and GPS antenna, which guaranteeds accuracy in the long term.

The 36 bit header timestamp is the number of half ms since some reference time. This reference time is chosen by setting the value of a certain register in the `timekeeper` block to be the current number of seconds since the reference time. There is enough bits to use 00:00:00 UTC of January 1st of the year of observation as the reference time. Once set, the firmware will keep track of the number of PPS pulses and clock cycles that have elapsed since the register was set. These are added to the value in the register to create an absolute header timestamp accurate to 0.5 ms. The 9 bit timestamps in the photon words (the number of µs since the last half ms) are relative to this frame header timestamp. Combining these timestamps gives an absolute timestamp for every photon precise to 1 µs.

Once photons are detected for each list of 256 channels, the photon detections are combined into one bus by the `pack_bus` block. This block contains four FIFO (first in

first out) buffer blocks. Each FIFO is read out once every four cycles. When a photon is detected in one of the four capture blocks it is written to a FIFO and waits until that FIFO is read.

Sending Photon Data

Figure 4.13 shows how the detected photon words are sent out of the FPGA. Two modes can be chosen by writing to certain registers. In phase dump mode, continuous phase samples from one chosen channel are sent out of the Virtex-6 via a one Gigabit ethernet port. The `phase_dmp` block in Figure 4.11 packs five consecutive phase samples from the chosen channel into a 64 bit word for sending. Long timestreams of phase data are useful for checking for slow shifts of baseline and for collecting many examples of photon pulses. Pulses are used in making optimal filters as mentioned in Section **??**.

The second mode is to send detected photon packets over the ethernet. The `gbe64` block collects some number of 64 bit words (usually 100) before generating an ethernet frame. This is sent as a UDP (user datagram protocol) frame. At the beginning of each frame, a 64 bit header word is pre-pended. The 8 most significant bits of the header are all ones, to identify this as a header. (The first eight bits of photon packets will not be all ones, until we have large enough arrays of MKIDs that we need the full 10 bits available for the x coordinate. When that happens another header scheme will need to be developed). The next 8 bits identify which ROACH2 is sending the frame. After that are a 12 bit frame number. This is useful for checking that the data acquisition computer is receiving all the UDP frames that the firmware is sending. The least significant 36 bits are the half ms timestamp from the `timekeeper` block.

When combining the 36 bit header timestamp and the 9 bit photon timestamp, care must be taken when the 9 bit µs timestamp loops back to 0 since the cycle when the header timestamp has been written to the ethernet buffer. In this case some multiple

of 0.5 ms needs to be added to photon timestamps depending on how many times it has looped by the time the photon was detected. The header is written to the ethernet buffer in the cycle after the last byte of data from the last frame has been written.

Inside the `gbe64` block, there is a CASPER yellow block for sending data out by ethernet. Data is written to the buffer in this block one byte at a time, until it receives an EOF (end of frame) signal. At this point is sends the data in the buffer. There is an option in the `gbe64` block to limit how long it waits to collect 100 photons. We generally set this to 0.5 ms. This ensures that even if there are very low count rates for the pixels processed by a particular ROACH2, the ROACH2 will not stop sending frames. The EOF signal sent to the yellow block has to be accompanied by a data byte to be sent. So, to force the block to send a frame when there is not a photon to send with the EOF, a fake photon word must be sent. This fake work consists of one zero (at the most significant bit) followed by sixty-three ones. This makes it easy to tell it apart from header words and real photon words. Most ethernet frames will have 808 bytes (in the data portion of the datagram), consisting of the eight byte header and 800 bytes of photon words. If a forced EOF (and fake photon) are sent, the frame size will be smaller. We could increase the typical frame size to make the transfer more efficient. (There would be less overhead due to UDP headers). For high data rates, we may need to consider this. However, frames larger than 1500 bytes are considered jumbo packets, and network configurations would have to be adjusted to allow for these.

The UDP protocol used here is much simpler than the TCP (transfer control protocol) used in ARCONS. On ARCONS, the FPGA would write photon packets into a yellow block BRAM, which had to be polled often by a C program running on the ROACH PowerPC. This program would send TCP packets to the data acquisition computer. TCP involves a number of handshaking signals that ensures two entities are connected and ready for each transfer. TCP will also check if each packet is received, and if not, will

resend the same packet. On the other hand, UDP has no handshaking. An entity sending UDP packets does not check if the other side is ready to listen. UDP packets are sent only once. There is no built in check for whether packets were successfully received. If they are not received, they are lost. In order to send high data rates with UDP without loss, I found the network configuration on the receiving end had to be adjusted. In particular the receive buffer had to be increased quite a bit from its default size.

4.3.3 Strategies for Timing Closure

In compiling the firmware for either the Virtex-6 or Virtex-7, Xilinx tools have to map all of the logic and operations of the design to specific resources available in the FPGA. It has to place and route paths between them in such a way that signal from each operation can get to the next operation in a fixed amount of time, often one cycle of the main fabric clock (4 ns for the DARKNESS Simulink design on Virtex-6). How long signals have to get from one place to another are determined by timing constraints. The CASPER libraries set period constraints for whatever clocks are involved in a design (internal and external). When compilation results in a placement in which all timing constraints are met, the design is said to achieve timing closure.

The Vivado tools for compiling Virtex-7 designs is good at achieving timing closure without low-level intervention from the firmware designer. The ISE tools for compiling Virtex-6 designs however, do require more detailed planning by the firmware designer to achieve timing closure. The resource utilization of the Virtex-7 ADC/DAC firmware is 42% of slice LUTs, 23% of slice registers, and 31% of BRAM. For the Virtex-6 channelization firmware, the utilization is 42% of slice LUTs, 28% of slice registers, and 48% of BRAM. More utilization makes timing closure more difficult.

For the Virtex-6 firmware it was necessary to guide the placement of resources by

adding location constraints. This involved constraining all logic inside of high-level Simulink blocks to occupy certain regions of the Virtex-6 chip. Figure 4.14 shows the placed resources color-coded by which high-level block they are part of. Unconstrained placement does not seem to take into account the large scale flow of data in the firmware. For instance, low-level logic inside a PFB FIR block does not need to be anywhere near logic in one of the `capture` blocks, because data has to flow through many blocks before it gets there. Using this information, I wrote constraints that start with placing the PFB FIRs in the northwestern corner of the chip, and lay out subsequent blocks in a counter-clockwise fashion. From Figure 4.14 it is clear that the FFT is the component that requires the most resources and occupies the most real estate on the chip. The design compiled for this figure did not include a full 50 taps in the programmable phase FIRs, which is why the northeastern corner of the chip does not look more congested.

Besides floorplanning with location constraints, there are a few other tricks to help achieve timing closure. One is to prevent one of the optimizations that the compilation tools have built in. Under certain circumstances, the placement tools will choose to use one SRL (shift register latch) for two different operations. This does not change the logic of the program, but it saves resources. Unfortunately this causing timing errors, because often a bit of logic in one part of the chip will combine with a bit of login in a far off part of the chip. Signals do not have time to travel to this far off place and get back. I had to make sure the circumstances in which the compilation would perform this optimization did not happen for spread out logic. (Usually this involved changing parameters in Xilinx delay and counter blocks to not be implemented in behavioral HDL).

Another common maneuver used to help timing is to add delays to critical paths that cannot make timing. Often in the firmware the duration of latency between operations is not important as long as it is consistent. This is why in Figures such as 4.10, two cycle delays are seen between each major subsystem block. Pipeline delay blocks which instan-

tiate some number of registers are often more helpful than normal Xilinx delay blocks with multi-cycle delays. The latter are usually instantiated with single shift registers. Adding register delays are also helpful when timing problems come from high fanout, which is when a single signal must be transported to many distant places in one clock cycle. Adding a tree of registers reduces the fanout.

4.4 Software

The software needed for ARCONS and DARKNESS are kept in github repositories.[1][2]

The main DAQ (data acquisition) computer connects to the PowerPC with a telnet session customized with the CASPER developed KATCP protocol, (Parsons et al., 2006). Katcp commands are wrapped in the `casperfpga`[3] python class. This python wrapper allows reading and writing to yellow block registers and memory (BRAM and QDR) in the Simulink design. All interaction with the Simulink firmware is done through yellow blocks.

The ADC/DAC board flash is programmed via a USB-JTAG (Joint Test Action Group) port using Vivado. After this the readout unit is disconnected from USB and inserted in the readout crate (See Figure 3.3). In the crate, all communication with the boards is done by ethernet, either to the PowerPC or directly from the ROACH2 for data transfer. Communication with the ADC/DAC board is done through the `a2g_ctrl` subsystem in the ROACH2 firmware.

When powered on, the ROACH2 PowerPCs boot their operating system (OS) from flash memory, and send a DHCP (Dynamic Host Configuration Protocol) request to the network for an IP (internet protocol) address. The data acquisition computer is set up

[1] ARCONS software and firmware is found at https://github.com/bmazin/SDR
[2] DARKNESS software and firmware is found at https://github.com/mstrader/MkidDigitalReadout
[3] https://github.com/ska-sa/casperfpga

as a DHCP server. It assigns each ROACH2 a static IP based on its ethernet hardware address.

When powered on, the ADC/DAC board loads firmware from flash memory and starts running a C program on its MicroBlaze. This program first initializes the various components of the board, mostly through SPI (Serial Peripheral Interface) buses. It also sends the sysref pulse to the ADCs and DAC to synchronize them. Then it enters a loop waiting for commands from the ROACH2 over the Z-DOK. Each numerical command is a single byte. Commands include:

- Initialize the UART interface
- Load the DAC LUT with data provided by the ROACH2
- Begin playing the DAC LUT to the DAC
- Initialize the LO in integer mode
- Initialize the LO in fractional mode
- Set the LO frequency to a provided value
- Initialize variable attenuators
- Set a variable attenuator to a given value
- Set ADCs into ramp calibration mode
- Unset the ramp calibration mode

Initializing the DAC LUT involves configuring the DMA to use hardcoded addresses in the DDR3 as the beginning and end of the LUT. The LO fractional mode has better frequency resolution than the integer and is generally used. Integer mode has better

stability, but both are more than sufficiently stable for our purposes. The ADCs can be configured to output a 12 bit ramp (sawtooth wave) instead of digitized data. We use this for calibrating the ROACH2 MMCM phase. When the MMCM phase is in a good range, the ramp comes across the Z-DOK perfectly and matches an expected model. If there are deviations from a perfect ramp, the MMCM phase is incremented until the ramp is perfect.

There are two python GUIs (graphical user interfaces) that use the `casperfpga` class to configure several readout units simultaneously. The first is `InitGui.py`. This programs the Virtex-6 with firmware, initializes communication between the Virtex-6 and Virtex-7, calibrates the MMCM phase, and calibrates the ROACH2 QDR memory interface.

Then the `HighTemplar.py` GUI can be run. This sets up the channelization process. Given a list of resonators for each readout unit, it generates DAC and DDS LUTs (with appropriate powers for each resonator) and loads them into the boards. It performs a sweep of the LO frequency to acquire I/Q loops of all resonators. It acquires phase timestream samples from each channel and sets an appropriate threshold.

Once everything is set up, photon data acquisition can be done from the `MkidDashboard.py` GUI. This program starts a C program called `PacketMaster`. `PacketMaster` receives UDP frames from all the Virtex-6's containing photon packets and writes them to disk. It also partially parses the photon packet stream to count photons in each pixel to generate one second images. These images are displayed on the dashboard.

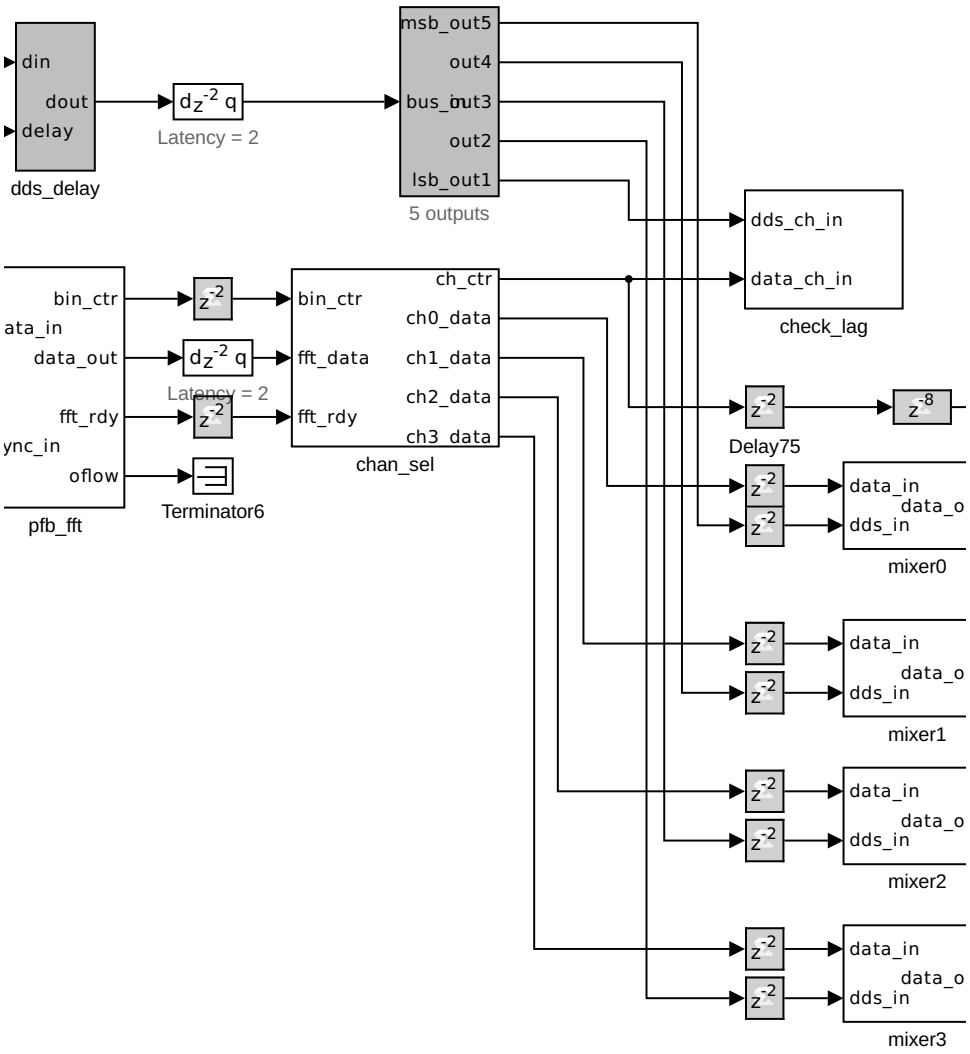

Figure 4.7: Section B from Figure 4.2. After coarse channelization by the FFT, some FFT bins are selected as channels (by the `chan_sel` block) and sent to four mixers to be multiplied with the DDS signal.

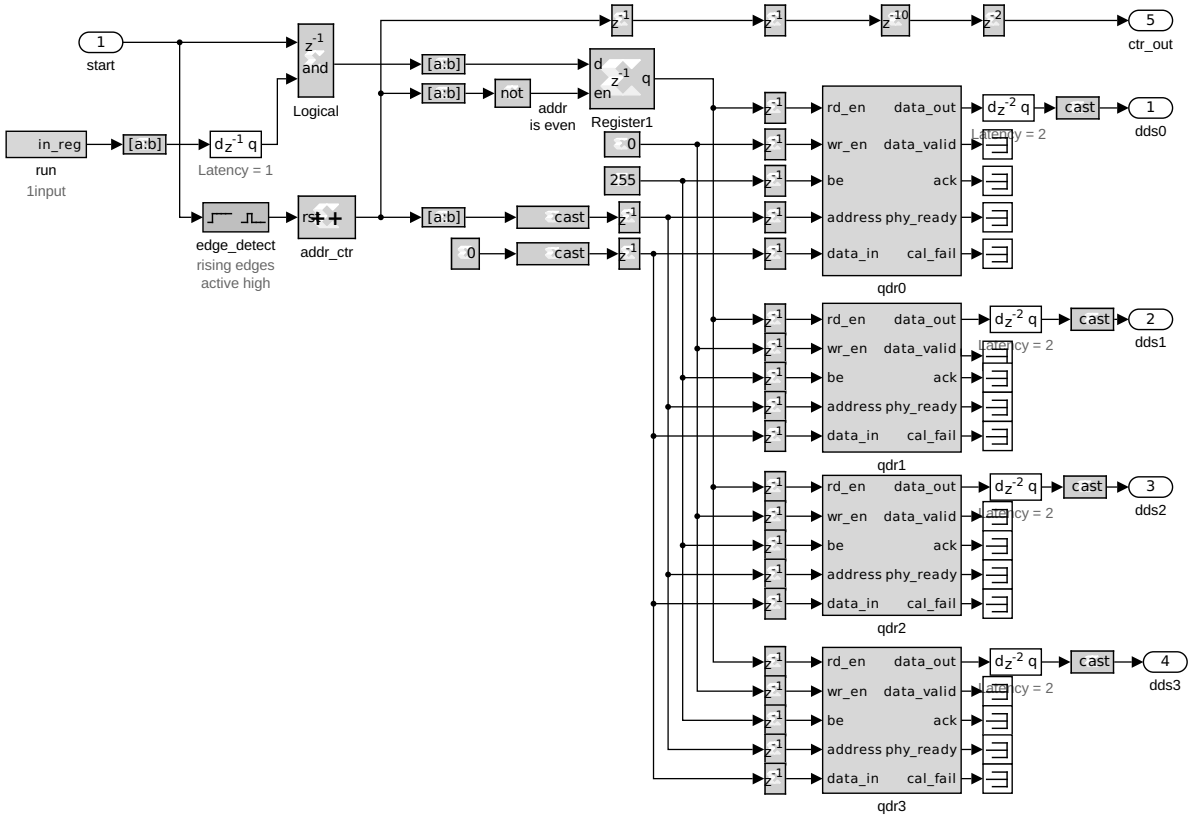

Figure 4.8: The inside of the `dds_lut` block in Figure 4.4. The LUT is stored in four QDR chips. They are read simultaneously.

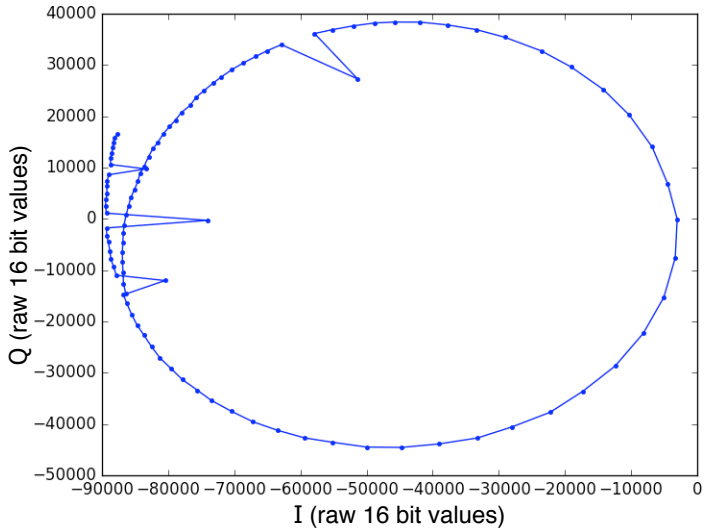

Figure 4.9: An example I/Q resonator loop for a single MKID. It was acquired by stepping the LO frequency such that a DAC tone frequency steps near the resonant frequency. At each frequency step averaged I/Q values are read from the `acc_iq` block. The I/Q point furthest to the right corresponds to the resonant frequency for the MKID. The anomalous points on the left were likely due to intermittent errors in programming the LO, which were later corrected.

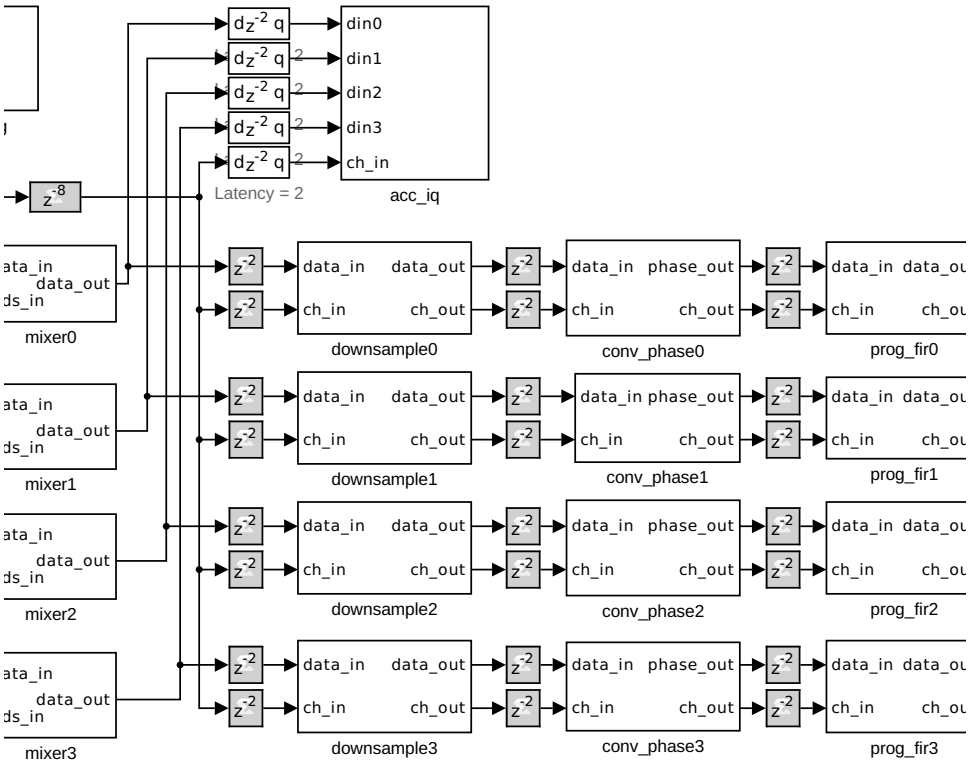

Figure 4.10: Section C from Figure 4.2. After FFT bins are mixed with the DDS, I and Q are passed through a 250 kHz filter which also downsamples to 1 MHz. Then the I/Q data is converted to phase. The phase is passed through programmable FIR filters. The `acc_iq` block on top can capture average I and Q values for all channels.

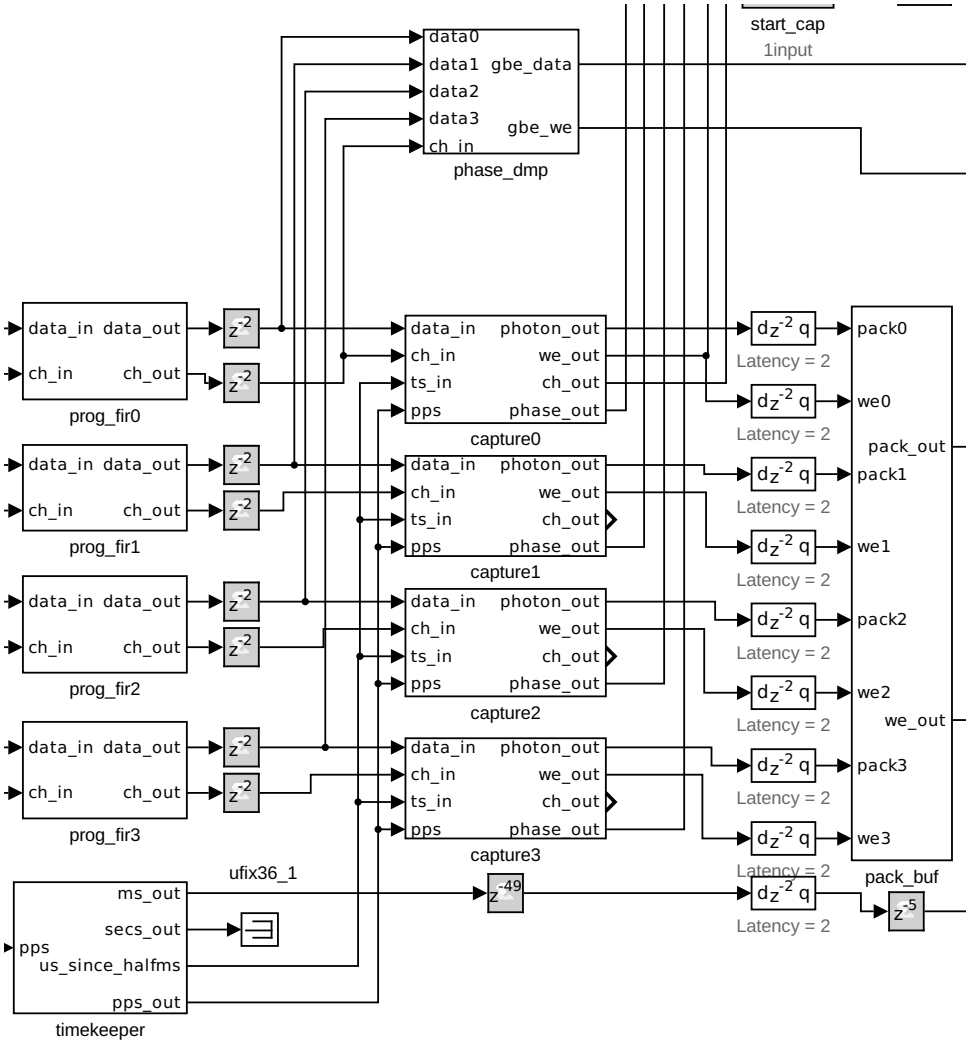

Figure 4.11: Section D from Figure 4.2. After the phase from all channels is filtered by the programmable filter, it is searched for pulses (in the capture blocks). Phase samples that meet the pulse trigger condistions are packed with timestamps and pixel data into a photon packet. The `pack_buf` block combines the four streams of photon packets into one. When `we_out` is True a photon has been found.

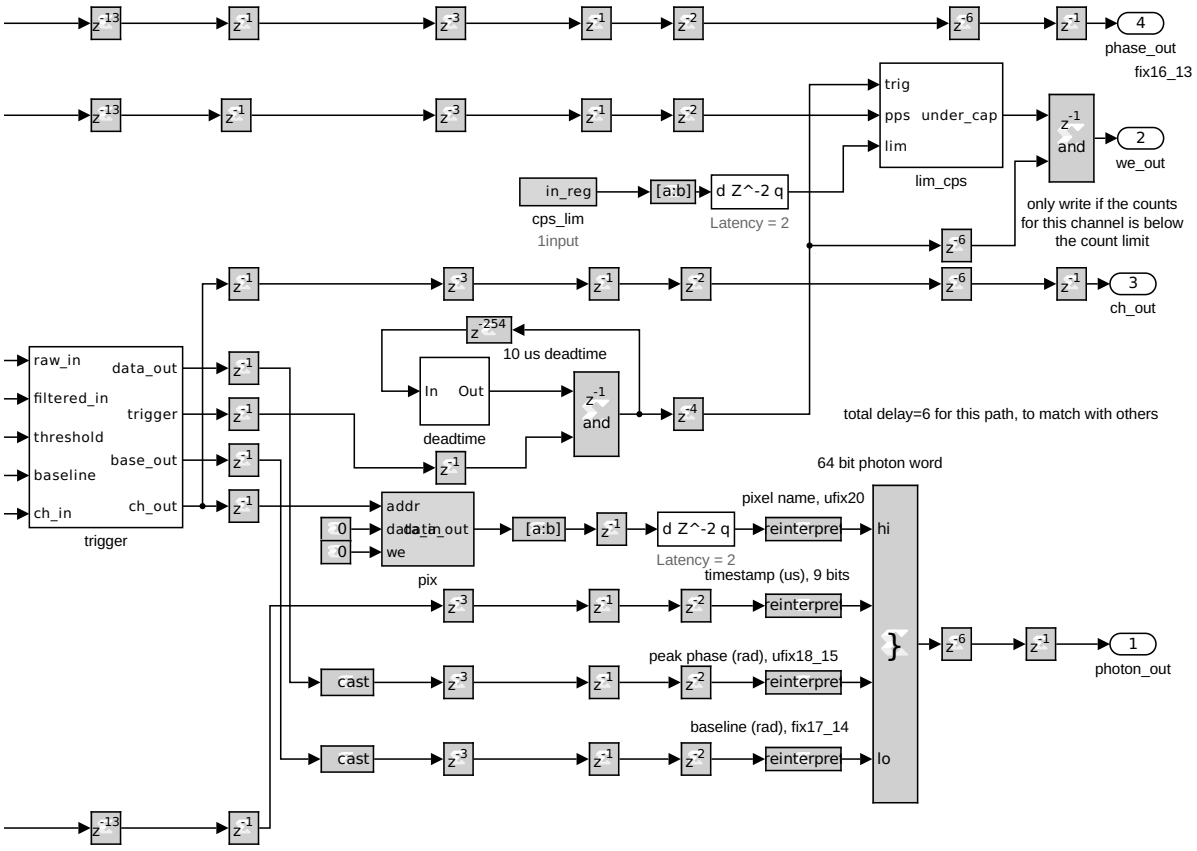

Figure 4.12: Part of the inside of the `capture0` block in Figure 4.11. After the SVF filter has determined the phase baseline for each channel and the thresholds have been loaded in, the phase can be checked for trigger conditions. When the conditions are met, a 64 bit photon packet is constructed and `we_out`=True.

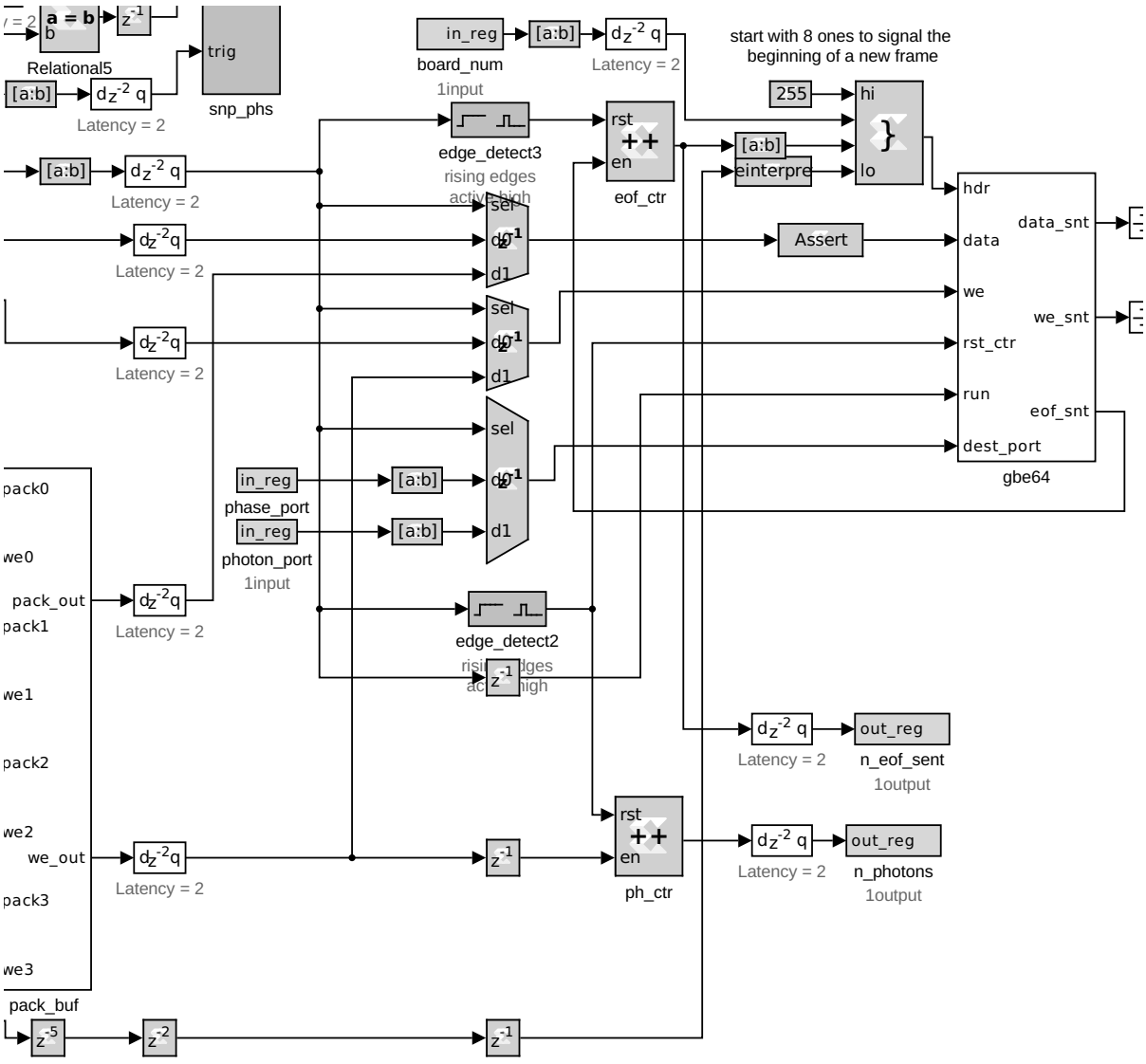

Figure 4.13: Section E from Figure 4.2. When the firmware is in capture mode, photons packets are sent to the 1 Gigabit ethernet (`gbe64`). Another mode allows phase to be streamed through ethernet. A header constructed here (by the block labelled "}" is also sent with every ethernet frame.

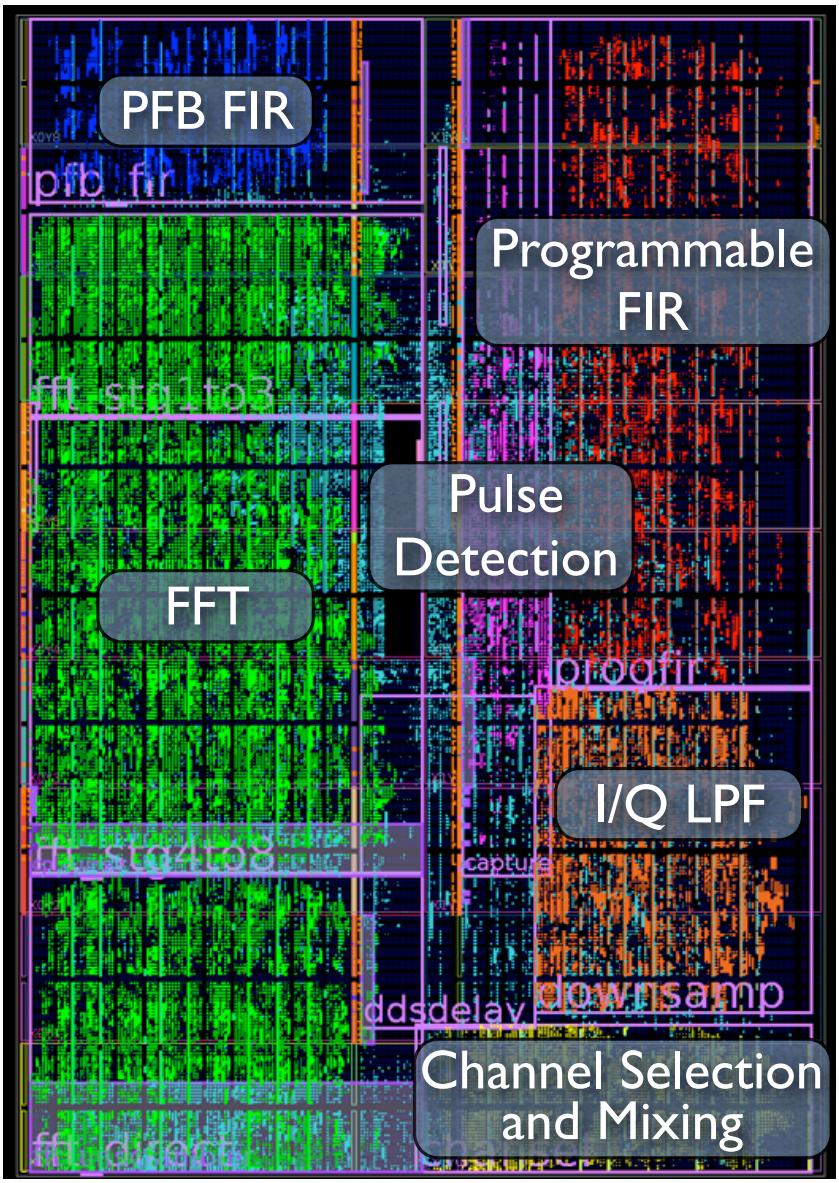

Figure 4.14: The floorplan of the Virtex-6, as seen in PlanAhead. Different colors are resources assigned for a particular set of blocks from the Simulink design.

Chapter 5

Characterization

5.1 Verifying the Channelization

I verified that the channelization algorithm discussed in Section 2.2 was successfully implemented in firmware, by replacing the ADC yellow block with a LUT made of yellow BRAM blocks. I could then send simulated ADC values (with very limited frequency resolution due to the length of the LUT) into the firmware and check that they are handled properly. Figure 5.1 shows measurements in firmware of the frequency response of a channel after coarse and fine channelizaton. I set up the DDS LUT for a chosen tone frequency (7.3 MHz in baseband, as it would be seen at the ADC.). I then filled the ADC simulation LUT with a series of tones around this frequency, and measured the power of the frequency that gets through the FFT and DDC at each step. The measured values differed from the theoretical value by a constant scale factor (a 4dB shift), due to the downshifts in the FFT. When adjusted for this scaling, the measured values agree well with the theoretical values.

I then inserted an exponential phase pulse into the DDS LUT for one tone. I verified that this pulse was recovered in the correct channel after channelization and converting

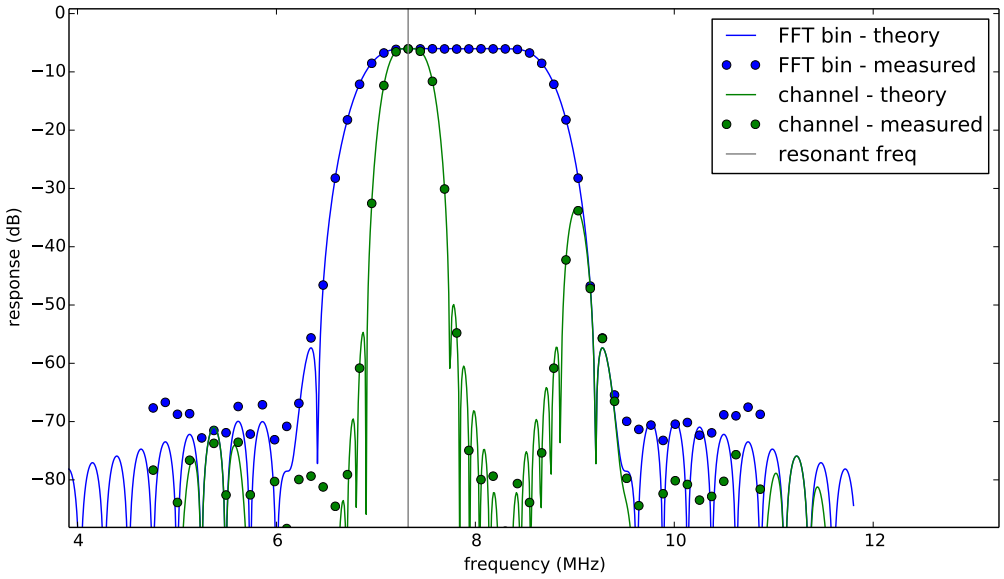

Figure 5.1: The frequency response function for one bin after coarse channelization (blue) and one channel after fine channelization (green) as predicted by theory (lines) and measured in firmware (circles).

to phase.

5.2 Verifying Tone Powers

To check that the power of the RF output of the system, we connected it to a signal analyzer, and generated a single tone with the DAC. The total attenuation of the RF variable attenuators was set to 35 dB, and the entire DAC dynamic range was used. The signal tone was seen on the signal analyzer at a power of −53 dBm, which is the expected power. The LO frequency was seen to leak into the RF output at a power of −64 dBm. The negative sideband of the generated frequency (the tone frequency mirrored over the LO) was seen as 30 dB below the signal tone. This is adequate sideband suppression by the IQ mixers, but may be improved by tuning the amplitudes and relative phase of the DAC LUT I and Q to cancel out small imbalances in the IQ mixers (See the procedure

performed in van Rantwijk et al. (2016) Section VI A.)

For the ADC side, we looped the RF output to input and probed a few spots on the RF/IF board before and after the amplifiers that lead to the IQ mixer using an oscilliscope. We found that the amplification was not as high as expected. This is currently being investigated at Fermilab.

5.3 Loopback Noise Tests

We characterized the noise of the readout system by looping the output of the RF ouput of a readout unit to the RF input. We could then take snapshots of I/Q values and phase in the firmware for chosen channels. In post-processing we computed the frequency spectrum of these timestreams. Figure 5.2 shows the frequency spectrum of a single channel I/Q data after coarse and fine channelization.

When generating a single tone with the DAC LUT, the noise spectrum appears as expected. However, when more tones are generated, each with the same power, noise lines appear at the DAC frequency resolution and its harmonics (See Figure 5.3). We suspect these are nonlinear mixing products from the IQ mixers on the RF/IF board. The power of these mixing products is much higher than expected. This problem will have to be investigated and resolved in the next revision of the readout before the next observing run of DARKNESS. The current phase noise makes it difficult to trigger on photons. For the DARKNESS commissioning run, our temporary workaround was to set very strict trigger thresholds. This triggered only on the largest photon pulses. This seemed to work, and allowed us to make our first light observations.

We also discovered a problem in the sideband suppression in the RF input chain into the ADC. As discussed above, the negative sideband of a tone generated above the LO frequency was suppressed by 30 dB. However, when this was mixed down and digitized

by the ADC, we saw much less suppression. In certain cases, the negative sideband was seen with the same power as the desired positive sideband. When a full set of tones at MKID resonant frequencies were generated, tones from the undesired sidebands would interfere with many of the desired tones. Our temporary solution was to identify MKIDs whose resonant frequencies were on opposite sides of the LO, such that reading out both would create "beater" pixels. For each pair found, only one pixel was chosen to read out. This decreased the fill factor of images captured by DARKNESS. This must also be investigated and addressed before the next observing run.

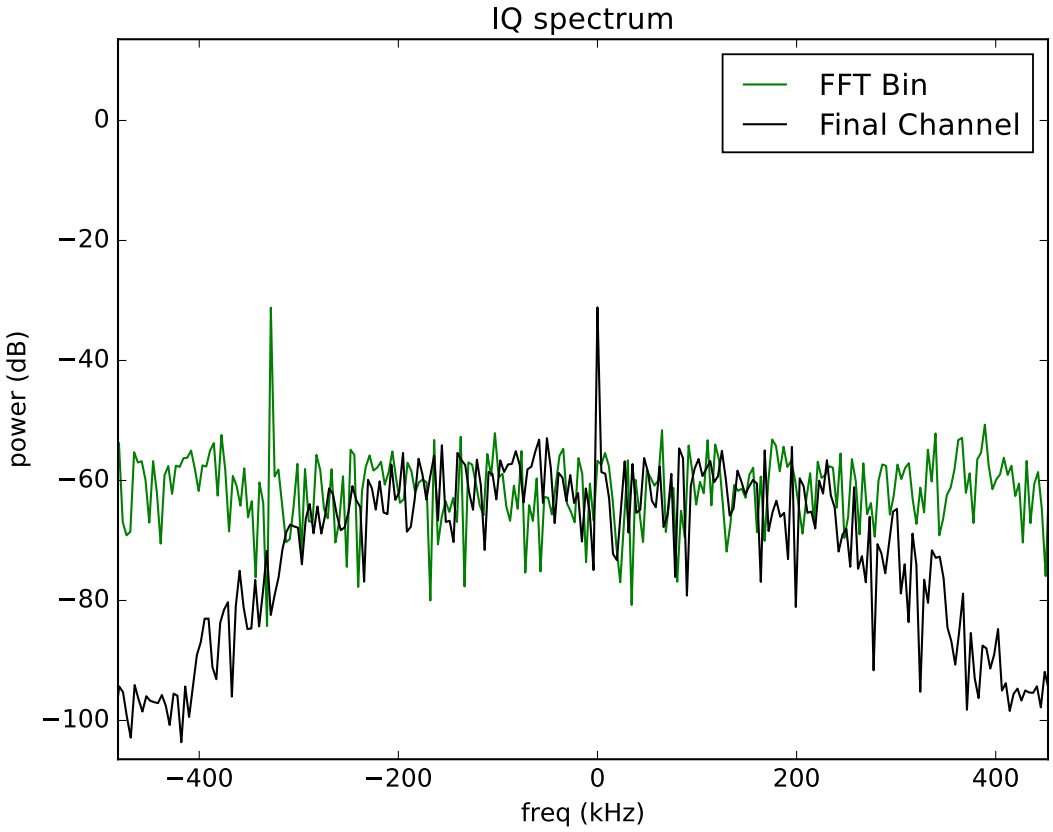

Figure 5.2: The frequency content of I/Q signals at different stages of channelization. The green line shows the frequency content of a selected FFT bin timestream. A tone frequency offset from 0 Hz is seen, because the original tone was not at the FFT bin center frequency. Mixing this tone with the DDS LUT shifts the tone to 0 Hz. Then the bandwidth is narrowed with a low pass filter to create the final channel (black). The signal tone is seen to be about 25 dB above the noise floor. For this test, only one tone was present in the DAC LUT.

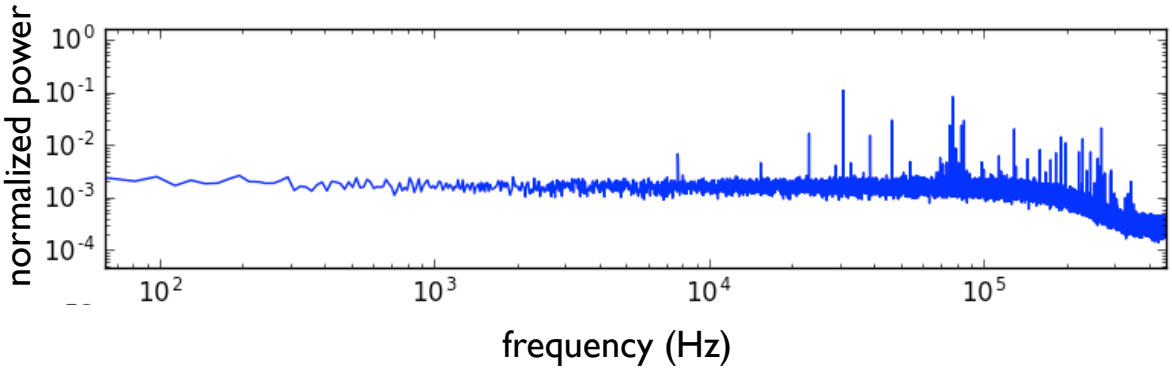

Figure 5.3: The phase noise spectrum for a single channel. For this test ten well spaced tones were in the DAC LUT. The RF signal was looped back from the RF/IF board output to the input without passing through the cryostat. Noise lines appear at 7.7 kHz and harmonics. The phase was not filtered. The rolloff at high frequencies is due to the 250 kHz I/Q filters in the last channelization step.

Chapter 6

Future Work

6.1 Debugging

The first order of business for future work on the digital readout is to fix the bugs covered in Chapter 5. With a little more investigation, the unexpected noise harmonics and flawed sideband suppression will likely be fixed in the next set of hardware and firmware revisions.

6.2 Features to Add

Without any changes to hardware, there are a number of ways to improve the system with tweaks to the firmware. For one, the programmable FIR applied to each channel may be improved. At the moment, it is barely possible to achieve timing closure with 50 taps in the FIR. The difficulty in timing seems to be a fanout problem. For 50 taps, there are one or two signals that is duplicated and sent to 50 places on the chip. The timing issues may be improved by changing the scripted block to make a tree of register delays for these paths to reduce fanout. If this works, more taps may be added to the

FIRs allowing for more sophisticated filters.

We have not yet tested the extended matched filters based on the work of Alpert et al. (2013) discussed in Section 2.3.1. We should determine if making matched filters orthogonal to nuisance vectors such as baseline or exponential tails can improve our energy resolution at high count rates.

Another feature likely to be added soon is to include photon energies in the photon packets sent out by the firmware to be written to disk. At the moment, the peak phase of each photon is saved. This is converted to wavelength in post-processing (See van Eyken et al. (2015) Section 4.3 for an explanation of how this conversion is done). Once we perform the phase to energy calibration, we could load the resulting conversion coefficients for each channel into some memory in the firmware and perform the multiplication there. This would make it simpler to create data cubes (two spatial dimensions and one energy) in real time at a frame rate of 2 kHz.

There are plans to use DARKNESS and MEC as both science cameras and wavefront sensors. By feeding back fast images or image cubes, we could perform closed-loop speckle nulling (Bottom et al., 2016; Martinache et al., 2014; Bottom, 2016).

6.3 Further in the Future

There are plans to build an MKID camera for the second phase of the PICTURE-C coronograph balloon mission. We will have to consider how best to make an MKID digital readout with power consumption and weight limits.

We have already begun talking about the next generation of MKID readout boards. We hope to place a larger FPGA on the ADC/DAC board and port the Virtex-6 channelization firmware to run on it, eliminating the need to have two FPGA boards per readout unit. Of course, whenever the next generation is made, ADCs are likely to be

faster, allowing for more bandwidth and more pixels read out per board.

Part II

Applications

Chapter 7

Observations of the Crab Pulsar

7.1 Introduction

7.1.1 The Crab Pulsar

The Crab pulsar is among the youngest known pulsars. Pulsed emission from the Crab has been observed at most spectral ranges detectable, from dekameter radio wavelengths to gamma-ray energies greater than 100 GeV (Mutel et al., 1974; Kuzmin et al., 2002; VERITAS Collaboration et al., 2011). A nearly-orthogonal rotator, it displays both a main pulse and a interpulse. The close alignment in time of the Crab pulsar's emission from radio to gamma energies (after removing the appropriate dispersive delay) suggests that a single emission region gives rise to the entire spectrum. The detection of pulsed emission above 100 GeV by VERITAS implies an emission altitude of at least 100 km above the neutron star's surface (VERITAS Collaboration et al., 2011). The optical lightcurve is broader in time than the radio, but shows nearly concurrent peaks corresponding to the main pulse and the interpulse.

Most remarkably, and unlike the majority of pulsars, its radio flux is dominated by

a small number of giant pulses, which regularly have flux densities a thousand times the mean radio emission (Heiles et al., 1970; Staelin & Sutton, 1970). The occurrence of these giant radio pulses (GRPs) is random, but they appear in a narrow range of pulse phase nearly coincident with the peaks of the main pulse and interpulse, although between 4-8.4 GHz, giant pulses also occur at two additional phases that are not near optical peaks. Giant pulses show variability on nanosecond timescales (Hankins et al., 2003) and appear to follow a power-law distribution in amplitude (with power law indices of roughly -2 to -3), with possible deviation at the highest amplitudes (Argyle & Gower, 1972; Mickaliger et al., 2012).

7.1.2 Multi-Frequency Observations

Pulsar emission at optical and X-ray wavelengths is commonly ascribed to incoherent synchrotron radiation (Harding et al., 2008). Radio photons play an essential role in catalyzing emission of this radiation (Petrova, 2009). This predicts a link between radio emission and higher energy emission. Accordingly, the few associations of emission in different spectral bands that can be observed have an important role in constraining models of pulsar emission. For example, Lommen et al. (2007) found that when the radio pulse of the Vela pulsar arrives early, there is an enhancement in the immediately following X-ray peak, although shot noise from the X-ray emission from the surrounding Vela-X supernova remnant led to increased uncertainty in the X-ray fluxes of individual pulses. Repeated attempts to detect correlations between the radio and gamma emission of the Crab giant pulses have not uncovered any statistically significant correlation (Lundgren et al., 1995; Bilous et al., 2011).

An optical-radio correlation in the Crab pulsar has been detected, however. Optical pulses have been seen to be 3% brighter when accompanied by a GRP (Shearer et al.,

2003; Collins et al., 2011). Such enhancement, combined with the lack of radio-gamma correlation, could reflect rapid (<10 μs) local density variations in the plasma stream, which would affect the coherent (radio) emission but not the incoherent (gamma) emission (Hankins et al., 2003).

We extend and refine measurements of this correlation with simultaneous observations of the Crab pulsar using the GUPPI backend (DuPlain et al., 2008) of the Robert C. Byrd Green Bank Telescope (GBT) and a new optical through near-infrared instrument, the ARray Camera for Optical to Near-IR Spectrophotometry (ARCONS) (Mazin et al., 2013), at the 200-inch Hale telescope.

7.2 Observations

7.2.1 Radio Data and Analysis

We observed the Crab pulsar for two hours on 12 December 2012 using the recently built Green Bank Ultimate Pulsar Processing Instrument (GUPPI) at the GBT.[1] GUPPI was used in 800 MHz coherent search mode centered on 1.5 GHz.

We developed a timing model for the observation using the software package TEMPO2 (Hobbs et al., 2006). However, we adopted the dispersion measure (DM) reported in the Jodrell Bank Crab pulsar monthly ephemeris (Lyne et al., 1993).[2] This DM has an associated uncertainty of 0.005 pc/cm^3, equivalent to a radio timing uncertainty of $\sim$10 μs, which is the dominant source of error in the radio side of our timing comparisons.

Next, we analyzed single-pulse properties using custom software that utilized libraries from PSRCHIVE (Hotan et al., 2004). To eliminate pulses that were contaminated by

[1]The National Radio Astronomy Observatory is a facility of the National Science Foundation operated under cooperative agreement by Associated Universities, Inc.

[2]http://www.jb.man.ac.uk/pulsar/crab.html

sporadic, impulsive interference, we excluded any pulse from analysis that had a maximal band-averaged intensity in the off-pulse region exceeding the off-pulse mean by greater than five times the standard deviation of all the off-pulse samples. Approximately 0.3% of pulses were eliminated on the basis of this criterion. For the remaining pulses, we tabulated the mean and maximum intensity of the main and interpulse phase ranges as well as the arrival phase of the respective points of maximum intensity. Subsequently, we used TEMPO2 to calculate the barycentric arrival time of each pulse.

To establish an approximate flux calibration, we utilize the Crab Nebula. At 1.5 GHz, the Crab Nebula is both bright and unresolved by the GBT. We therefore estimate a contribution of $955\nu_{\mathrm{GHz}}^{-0.27}$ Jy ≈ 850 Jy (Bietenholz et al., 1997), which easily dominates the system noise. We thus equate this contribution with the average of the off-pulse region to obtain an approximate conversion from our measured flux density to Jy.

Observed radio pulses that had a peak flux density above 28.4 Jy and that arrived during the phase range 0.9915 to 1.0042 (where the phase of the peak of the average radio main pulse has been defined as 1) were tagged as main pulse GRPs. There were 7205 of these observed. Radio pulses with a flux density above 25.1 Jy that arrived in the phase range 0.3928 to 0.4099 were tagged as interpulse GRPs. In addition, interpulse GRPs that occur in the same period as a main pulse GRP were excluded, so that enhancement by main pulse GRPs does not affect the interpulse result. There were 2237 pulse detections meeting these requirements. The flux cutoffs were chosen to exclude the significant number of false positives at lower fluxes, which would artificially lower the average optical enhancement calculated, as discussed in Section 7.3.3. The phase constraints were chosen to include nearly all GRP detections while minimizing the number of false GRP detections. Outside of the chosen phase ranges, the fraction of false positives becomes significant and, similar to low flux GRPs mentioned above, the enhancement computed at these phases would be artificially lowered. Even with the flux and phase restrictions

roughly 3% of the pulses tagged as main pulse GRPs and 60% of the interpulses are expected to be false detections.

7.2.2 Optical Data and Analysis

The Crab pulsar was observed in the optical from the Coudé focus of the 200-inch Hale Telescope at Palomar Observatory with ARCONS from 03:30 to 06:37 AM on 12 December 2012 UTC. The seeing fluctuated between 1" and 1.5" during the observation.

The optical data were compiled into photon lists by the ARCONS pipeline (Mazin et al., 2013). Each photon was tagged during the observation with a timestamp derived from a Stanford Research FS725 Rubidium 10 MHz frequency standard, which was synced with the 1 pulse per second (PPS) output of a Meinberg GPS170PEX GPS board. The timestamps were later corrected for a 41 μs delay in the digital readout system, measured by triggering an LED with the PPS signal. Barycentering was done with TEMPO2, using a custom plug-in that allows for the analysis of individual photons.

The photons within a circular aperture (with a radius of five pixels) around the center of the pulsar's point spread function (PSF) were folded into pulse profiles, each with 250 phase bins, for GRP-accompanied optical pulses and their surrounding (40 pulses before and after) non-GRP-accompanied pulses. The standard deviation of the flux of the surrounding non-GRP-accompanied pulses was used to estimate the error in the GRP-accompanied profile. The sky level in the profiles was determined by the average of five phase bins in the off-pulse phase region and then subtracted. Following Shearer et al. (2003), the optical flux enhancement was calculated as the percent increase in the number of photons in the three phase bins around the main peak.

Spectra of the peak of the optical main pulse were made separately for GRP-accompanied and non-GRP-accompanied pulses. We excluded the 20-30% of pixels that did not have

a reliable wavelength calibration in the range from 4000 to 11000 Å (as determined by the wavelength calibration module of the ARCONS pipeline). The flat-field and spectral shape calibrations discussed in Mazin et al. (2013) were not used for this work, so the spectra calculated are not corrected for the wavelength dependence in each MKID's quantum efficiency. It is valid to compare spectra determined in this way, but the absolute spectral shape is not fully calibrated. The sky spectrum was also determined as the spectrum of pixels in a half-annulus around the pulsar's PSF, where only half an annulus was used to avoid contamination from the Crab pulsar's stellar neighbor. The resulting sky spectrum was subtracted from all pulsar spectra.

7.3 Results

7.3.1 Optical-Radio Lag and Optical Enhancement

We observe the optical pulse to lead the radio pulse by 202 ± 36 μs. This was determined as the time difference between the highest flux bins in the optical pulse profile (with 500 phase bins instead of our usual 250) and the radio pulse profile. The uncertainty comes from the optical phase half-bin width, which is 0.001 in phase or 33.6 μs, the radio phase half-bin width, which is 0.00024 in phase or 8.2 μs, and the uncertainty in the radio timing from the dispersion measure (DM), which is $\sim$10 μs. The overall uncertainty was estimated as these three uncertainties added in quadrature. Our measured optical-radio lag time is comparable to the recent measurements of Słowikowska et al. (2009) and Oosterbroek et al. (2008) at 231$\pm$68 μs and 255$\pm$21 μs, respectively.

We found that when an optical pulse is accompanied by a main pulse GRP, the peak of the optical main pulse is enhanced, on average, by 3.2%$\pm$0.5% with a significance of 7.2 σ, as shown in Figure 7.1 and Figure 7.2. Interpulse GRPs were observed to correlate

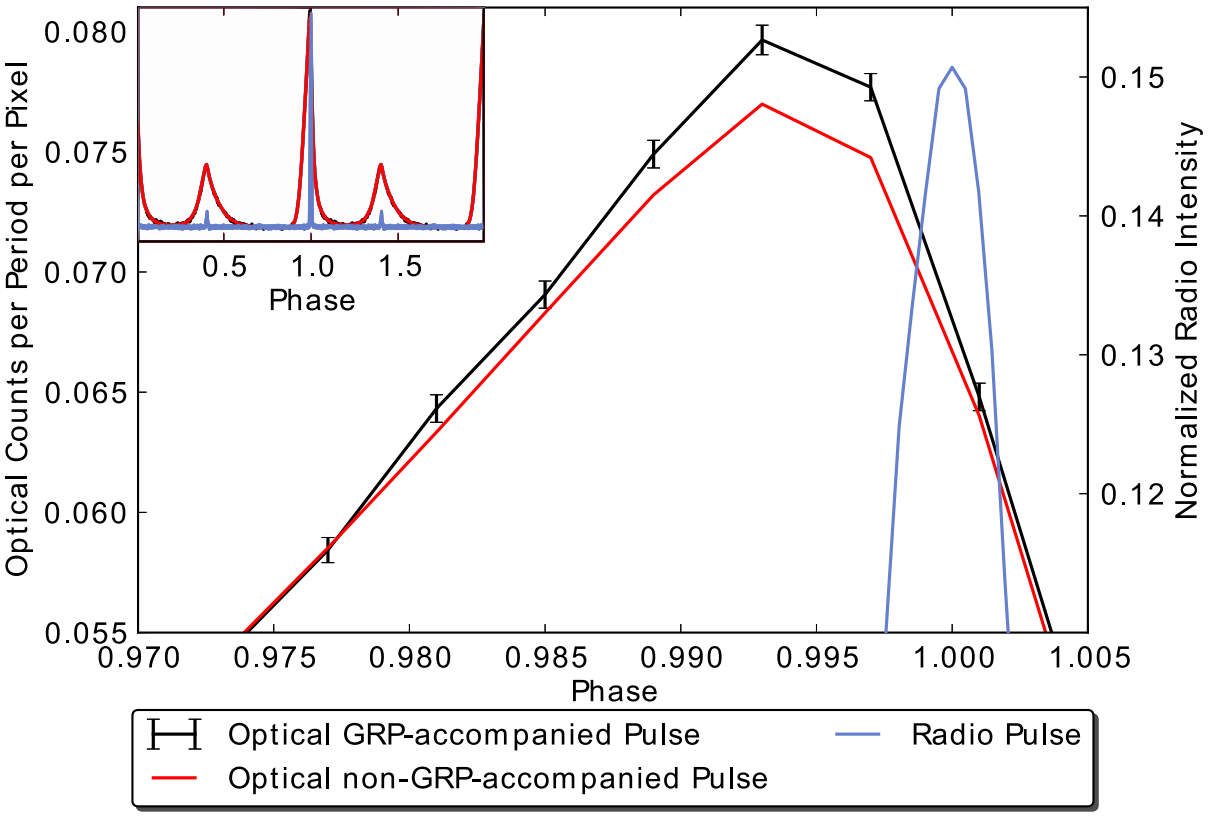

Figure 7.1: Average optical profile of the peak of the main pulse for 7205 pulses accompanied by a main pulse GRP (black) and for 80 pulses surrounding each of the 7205 accompanied pulses (excluding other GRP-accompanied pulses) (red). The normalized radio pulse profile is also shown (blue). The inset shows two full pulse periods displaying the main pulses and the smaller interpulses in both optical (red) and radio (blue).

with a moderately significant optical enhancement of the peak of the optical main pulse as well. This enhancement was measured to be 2.8±0.8% with a significance of 3.5 σ. An enhancement of this kind was also observed by Shearer et al. (2003) at 1.75 σ. Our result is surprising given the expected number of false detections in the interpulse data. Future observations with higher signal to noise ratio may find higher enhancement.

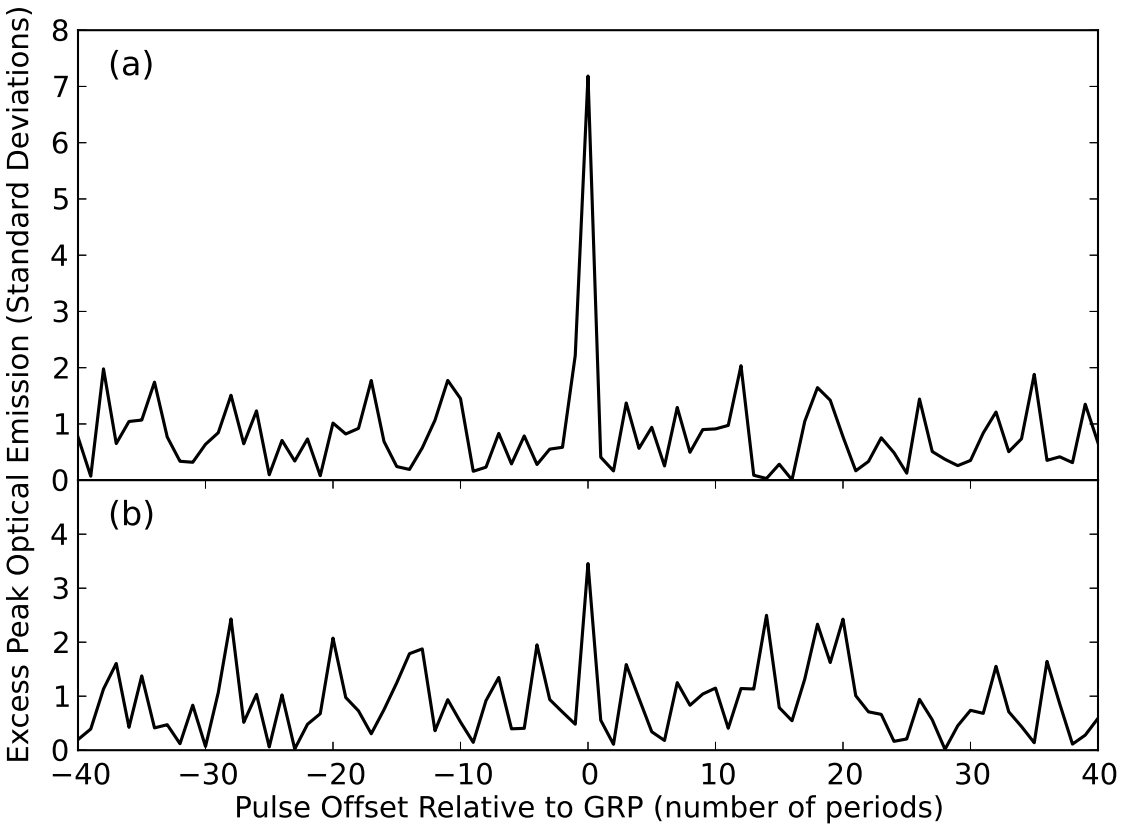

Figure 7.2: The number of standard deviations between the peak flux, for pulses with a fixed offset from a GRP, and the mean peak flux calculated for (a) main pulse GRPs and (b) interpulse GRPs. Mean flux is an average over all pulses within 40 pulses of a GRP. The peak flux is defined as the sum of the three phase bins around the peak of the main pulse.

7.3.2 Radio Phase Correlations

We found a correlation between optical enhancement and the phase of the accompanying main pulse GRP peak (Fig 7.3). The optical enhancement appears to be higher for certain arrival phases, with a maximum of 11.3±2.5% for the phase bin with range 0.9934 to 0.9944, which is where the peak of the optical main pulse is. To determine the significance of the observed enhancements diverging from a constant value, we performed a $\Delta\chi^2$ test. In a similar application this test is sometimes used in the transit community to determine if a small eclipse is present in a light curve (Siverd et al., 2012; Fressin et al., 2011). For the statistical test we use finer binning (bin widths are 0.0005 of a period). We first fit a flat line model, resulting in a level of 3.22±0.46%. This fit had a χ^2 of 40.5 with 25 degrees of freedom. We then fit a Gaussian model with four parameters, amplitude, sigma, x-offset, and y-offset, which took on the values 8.3±2.4%, 0.00075±0.00025, 0.99442±0.00025, and 2.54±0.54%, respectively. This resulted in a χ^2 of 23.9 with 22 degrees of freedom. Comparing the two fits gives a $\Delta\chi^2$ of 16.6 with 3 fewer degrees of freedom for the Gaussian model. This difference corresponds to a p-value of 0.00084, indicating that the Gaussian model better represents the enhancement vs. phase data with a significance of 3.3 σ.

7.3.3 Flux Correlations

The results of probing for correlations between enhancement of optical pulses with an accompanying main pulse GRP and the flux of the GRP are shown in Figure 7.4. In this plot, the optical enhancement appears to be lower for less energetic GRPs, but this is likely due to the treatment of spurious radio signals as GRPs. Since spurious radio signals are not expected to have the enhancement associated with genuine GRPs and are more numerous at low flux densities, the calculated average enhancement is artificially

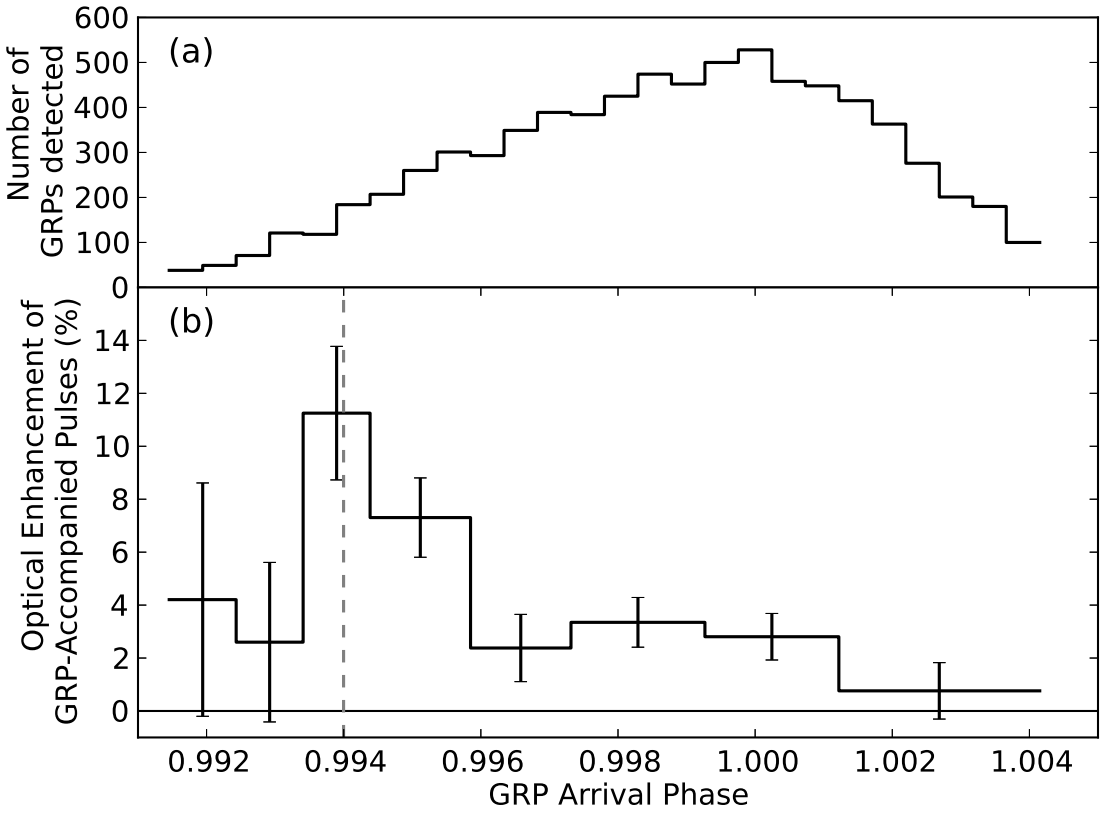

Figure 7.3: (a) Histogram of arrival phases for main pulse GRP detections. The expected number of false positives is about 8 per arrival phase bin. (b) Optical enhancement as a function of the GRP arrival phase. For reference, the phase of the optical main pulse at phase 0.994 is shown.

lowered for lower flux bins. The fraction of false counts was estimated as the ratio of the number of false GRP detections in an off-pulse phase range to the number of detections in the main pulse range. Using this, Figure 7.4 shows the expected enhancement curve if the actual enhancement is a constant of value 3.2%. Comparing the data curve with the predicted curve gives a χ^2 of 9.5 with 10 degrees of freedom and corresponding p-value of 0.48, equivalent to 0.70 σ significance in the difference between the data curve and predicted curve. Consequently, we find no significant change in the optical enhancement as a function of the flux density of the GRP.

7.3.4 Spectral Dependence

We found no statistically significant differences between the spectrum of the peak of the main pulse for enhanced optical pulses accompanied by a main pulse GRP and the spectrum for non-GRP-accompanied pulses (See Figure 7.5). We performed a two-sample Kolmogorov-Smirnov (K-S) test to compare the spectral distribution of photons in GRP-accompanied and non-GRP-accompanied pulses with the null hypothesis that the photon wavelengths are drawn from the same spectral distribution in both cases. The result was a D-metric of 0.0030 with a corresponding p-value of 0.41. So, any difference between the spectra has a significance of only 0.83 σ.

7.4 Discussion

The apparent match between the spectrum of ordinary pulses and GRP-enhanced optical pulses gives us a clue about the nature of the relationship between the radio and optical emission of the pulsar. In order to produce the same spectrum, the mechanism that generates the extra optical emission during an enhanced pulse is likely the same mechanism that generates ordinary optical emission.

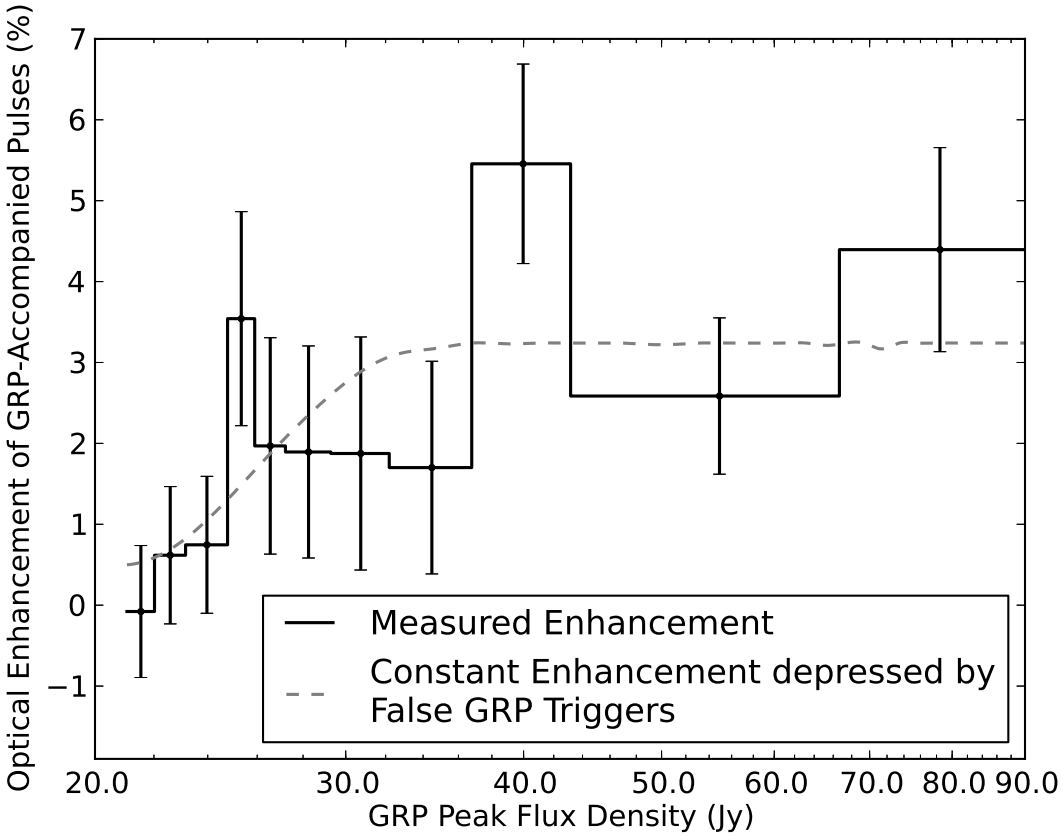

Figure 7.4: Enhancement of optical pulses accompanied by main pulse GRPs as a function of the peak flux density of the GRP (black). The fraction of spurious radio peak detections increases as detected radio pulses become weaker. The gray dashed line shows the predicted optical enhancement with the assumptions that optical pulses accompanied by false GRP detections have zero enhancement and that optical pulses accompanied by real GRPs have a 3.2% enhancement that is constant with respect to GRP flux.

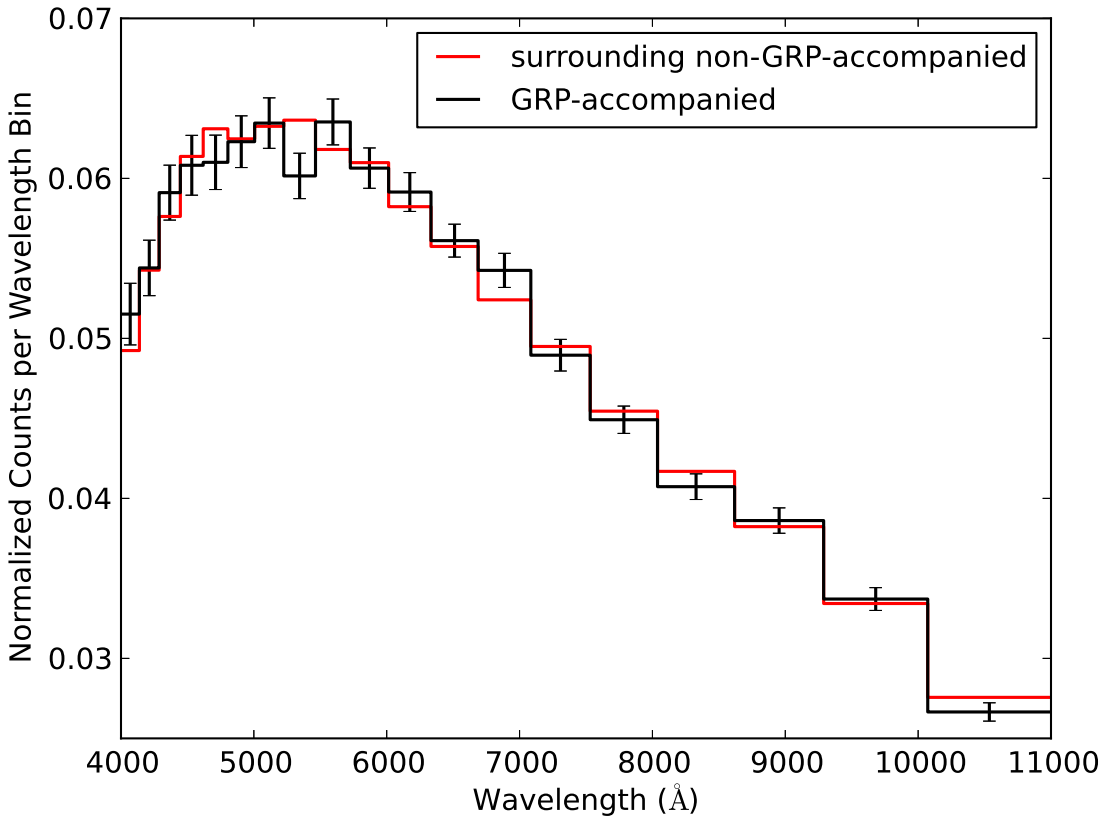

Figure 7.5: Spectrum of photons arriving during the peak of the optical main pulse (3 phase bins with highest counts) for GRP-accompanied pulses (black) and surrounding pulses (red) normalized to have the same integrated flux. The wavelength resolution has been oversampled. There do not appear to be significant spectral differences between enhanced GRP-accompanied pulses and non-GRP-accompanied pulses.

Interestingly, around the arrival phase for which GRPs were correlated with higher enhancements of the peak of the optical main pulse, Słowikowska et al. (2009) found that the slope of the average optical polarization angle changes rapidly.

The fact that the degree of optical enhancement depends on the phase of the corresponding GRP for the Crab pulsar strengthens the link between optical and radio emission. Shklovsky (1970) proposed that radio photons catalyze optical emission from pulsars. He noted that the large electric fields generated by the rotation of the pulsar's magnetic field easily accelerate electrons and positrons to high energies along field lines, but cannot easily contribute momentum perpendicular to the enormous B-field of the star. However, in the frame of a relativistic particle, a radio photon may be blueshifted to the frequency of the cyclotron resonance, and convert some of the particle's momentum along a magnetic field line to the transverse direction. The particle then emits synchrotron radiation at optical to X-ray energies. The spectral dissection of Harding et al. (2008) ascribes spectral emission of the Crab pulsar at these energies to synchrotron emission, and notes the role of catalysis by radio photons.

Shklovsky's picture would suggest that the stimulating radio pulse should arrive with the resulting optical emission. We observe this sequence for the most enhanced optical pulses, but often the radio pulse arrives later than the enhanced optical pulse, suggesting that a less direct mechanism is at work. The radio pulse may suffer significant delay due to magnetospheric refraction (Barnard & Arons, 1986; Lyutikov & Parikh, 2000; Jessner et al., 2010; Beskin & Philippov, 2012). Alternatively, if optical and radio emission were beamed in slightly different directions, they would appear at different pulse phase. However, this would require the duration of GRPs to be at least as long as the optical-radio lag. This duration would be difficult to reconcile with the discovery by Hankins et al. (2003) that GRPs are made of nanosecond bursts separated by microseconds.

Chapter 8

Observations of a Millisecond Pulsar PSR J0337

8.1 Introduction

Millisecond pulsars (MSPs) are distinguished from classical rotation-powered pulsars (RPP) in their characteristic parameters. MSPs have shorter periods, smaller period derivatives, and weaker magnetic fields. They are thought to be classical pulsars that have been spun up by accretion from a binary companion (Alpar et al., 1982; Bhattacharya & van den Heuvel, 1991). Many of the known MSPs reside in binary systems, generally with a low mass white dwarf companion.

The optical emission properties of MSPs are completely unknown, though other types of pulsars are seen in the optical band. Optical emission can be thermal, originating from the neutron star surface, or non-thermal, originating from particles accelerated in the magnetosphere or from X-ray reprocessing in a circumstellar debris disk. There are about thirty optical, ultraviolet (UV), or near-infrared identifications of isolated neutron stars. Of the RPPs, pulse timing in these bands has been achieved for only five objects

(Mignani et al., 2011).

White dwarf companions dominate any optical emission in a binary system, so most searches for MSP optical emission have focused on deep imaging of isolated MSPs. These attempts at deep imaging have not resulted in any definitive optical detections (Sutaria et al., 2003; Mignani et al., 2003; Koptsevich et al., 2003; Mignani et al., 2004), although Sutaria et al. (2003) did find a possible $m_V \approx 25\,\mathrm{mag}$ counterpart to PSR J1024-0719. Also, Kargaltsev et al. (2004) were able to observe UV emission from the neutron star in the spectrum of the binary MSP J0437-4715. This UV emission was ascribed to thermal emission.

Another approach to find optical counterparts of MSPs is to search for pulses, using a known rotational period from observations in radio, X-rays, or γ-rays. This can be done for pulsars in binary systems as well as for isolated pulsars. Optical pulses have not yet been found in any MSP. Optical timing is a challenge for millisecond pulsars due to their intrinsic faintness in the optical band and high pulsation frequencies. To perform this measurement, detectors need high time resolution and high quantum efficiency. There are just a few attempts in the literature to search for pulsed optical emission from MSPs, in particular, PSR B1937+21 and several MSPs in compact globular clusters, all with negative results (Manchester et al., 1984; Middleditch et al., 1988).

8.1.1 PSR J0337+1715

PSR J0337+1715 (hereafter J0337) is an MSP that resides in a hierarchical triple system with two white dwarves orbiting it (Ransom et al., 2014). The white dwarves have masses 0.2 $M_\odot$ (inner) and 0.4 $M_\odot$ (outer), with orbital periods 1.6 days and 327 days, respectively. Spectroscopy of the inner white dwarf has revealed that it has a surprisingly high temperature of 15800 K (Kaplan et al., 2014). Discovery of the system

has generated interest in using the system to test the strong equivalence principle of general relativity and to study the secular evolution of multi-body systems (Ransom et al., 2014).

For our purposes, the 18th magnitude inner white dwarf makes J0337 a convenient target for observing with our instrument, the ARray Camera for Optical to Near-IR Spectrophotometry (ARCONS; Mazin et al. 2013). The white dwarf allows us to see the system in one second integration images, so that we can check in real time that it is placed on the science array away from areas of dead pixels. The downside of course, is that the light from the white dwarf adds to the background noise, limiting the signal-to-noise ratio (SNR) that we could reach in our allotted time.

8.2 Observations and Analysis

We observed J0337 from the 200-inch Hale Telescope for a total of 6.7 hours on the nights of September 24, September 25, and October 21, 2014. Conditions were not photometric on the last night. Seeing during the observations generally ranged from 0".8 to 1".5. The seeing briefly went up to 2" during the last night of observation.

The ARCONS data reduction pipeline was used to extract photon lists from the raw data files (van Eyken et al., 2015). Each photon has an associated wavelength and timestamp. For simplicity and to avoid any unforeseen artifacts, flat and spectral response calibration weights were not applied. This means that the absolute measurement of flux in this analysis is not completely reliable. For this work, what is important is the flux of the pulsar relative to the inner white dwarf, so this does not present a problem. We used the pulsar timing software package TEMPO[1] to barycenter the photon timestamps using the JPL DE405 ephemeris and to fold them by the 2.7 ms pulse period. Also, part

[1]http://www.atnf.csiro.au/research/pulsar/tempo.

of the PRESTO package was used to parse TEMPO output files (Ransom, 2011).

To predict the pulsar rotational phase for each photon, we use the basic timing model from Ransom et al. (2014). This model constructs the Newtonian equations of motion as differential equations and solves them for the motions of the three bodies in the J0337 system. The parameters in Ransom et al. (2014) were obtained by fitting the predicted pulse arrival times to 26280 radio pulse arrival time measurements spanning 506 days, obtaining a residual scatter of 1.34 μs. In other words, this model is capable of predicting pulse arrival times to microsecond accuracy. We updated the model by including additional observations up to 2014 October and re-fitting.

The raw data from ARCONS are divided into five minute observation files with a gap of a few seconds between observations. Images were made from each five minute file and were fit with a circular Gaussian point spread function (PSF) to determine an optimal aperture size for each time period. The aperture radius was set to 1.6 times the Gaussian sigma of each PSF fit. Due to the uncertainty in the fits, setting the aperture size as a function of time in this way was somewhat noisy, so the list of aperture radii was first smoothed with an eight-point boxcar filter.

After this the circular apertures were applied to make a list of photons recorded within the PSF of the target object during all the observation times. Some segments of observation time were cut, because either they were taken while the seeing had inflated to 2", the object had drifted onto a section of dead pixels, or the object had been suddenly moved during observation. While observing, we view real time images from the science camera with one second integration times. Sometimes the target object was difficult to see in these images. Also, at the Coudé focus, the field would slowly rotate over time. Periodically, we would move the telescope by a few arcseconds and then back to ensure that the target was on the intended part of the array. With the high time resolution of ARCONS these sudden moves could be accounted for by tracking the PSF as it moved

and adjusting the aperture position accordingly, but the amount of data affected by this was small enough that it was simpler to discard it. The portions of data with worse seeing were discarded because as the PSF is spread over more pixels the ratio of signal from the neutron star to background sky counts decreases.

8.3 Results

Once the list of photons with corresponding wavelengths and timestamps was compiled, statistical tests could be applied to test for periodicity under a given wavelength cut. ARCONS has a broad wavelength sensitivity of 4000 to 11000 Å, but the SNR in the infrared is likely lower, so some wavelength cut was needed to optimize our search for pulsed emission. We tried to remove the arbitrariness of cutting at a particular wavelength range by testing for periodicity in many ranges, using the H test (de Jager et al., 1986). In this test higher H values indicate more significant periodicity. The results of the wavelength search are shown in Figure 8.1. The highest H metric had a value of 12.3 for the narrow wavelength range 3950 to 4350 Å. To find the significance of this H value, we created 1000 simulated photon lists with the same wavelengths as the real list, but with random pulse phases. We then applied the H test with the same set of wavelength cuts. We found that for 40.2% of the simulated lists, there was one or more wavelength ranges that generated an H metric at least as high as 12.3. This implies that finding an H value of 12.3 in the wavelength search had a low significance of 0.84 σ.

For the purpose of calculating an upper limit to the pulsed emission, we chose to use the wavelength range of 4000 to 5500 Å to approximately match the band pass of the Sloan Digital Sky Survey (SDSS) g filter (Doi et al., 2010). This allows for a rough comparison to the SDSS measured magnitude for the inner white dwarf. The binned, folded light curve for the three nights of observation in the wavelength range 4000 to

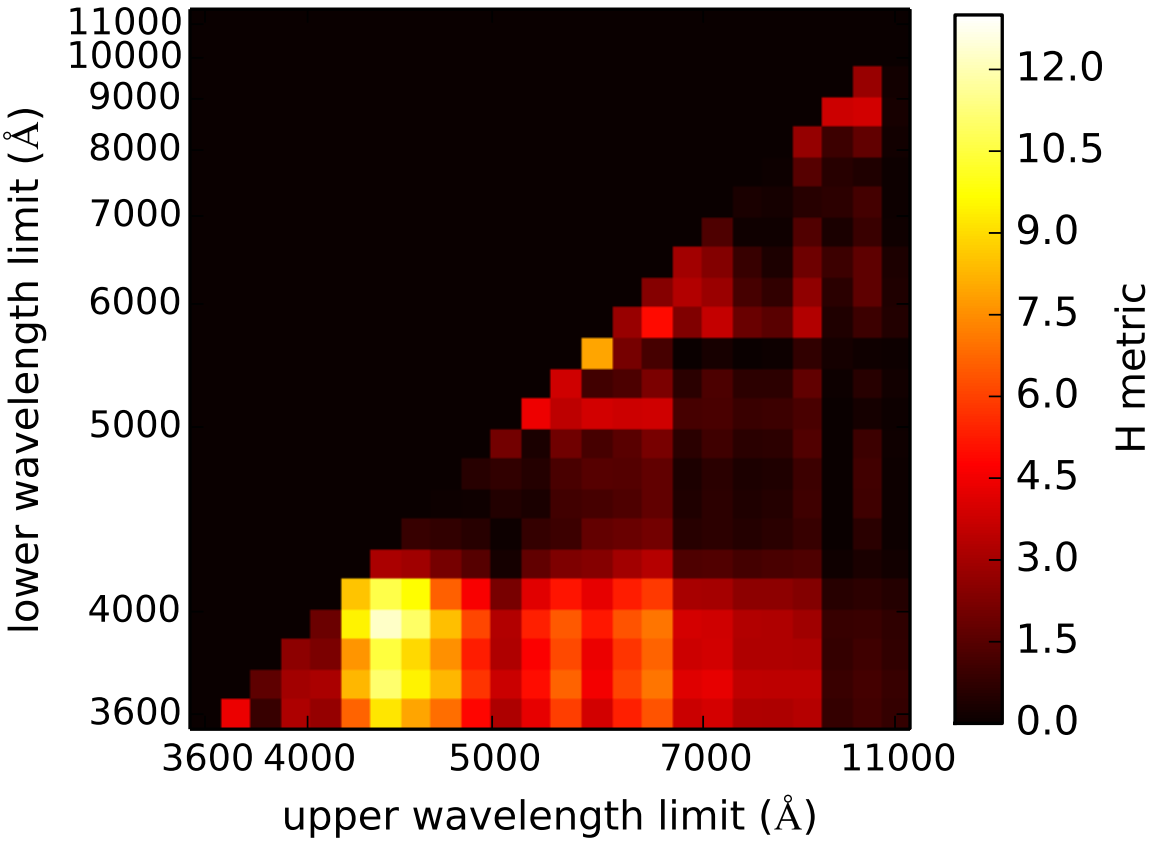

Figure 8.1: A search for periodicty with many wavelength cuts. The color axis gives the metrics produced by applying the H test to the photon timestamps with various wavelength cuts. The narrow wavelength range 3950 to 4350 Å had the highest H metric value of those tested (indicating a higher probability of periodicity), but the value obtained was not statistically significant.

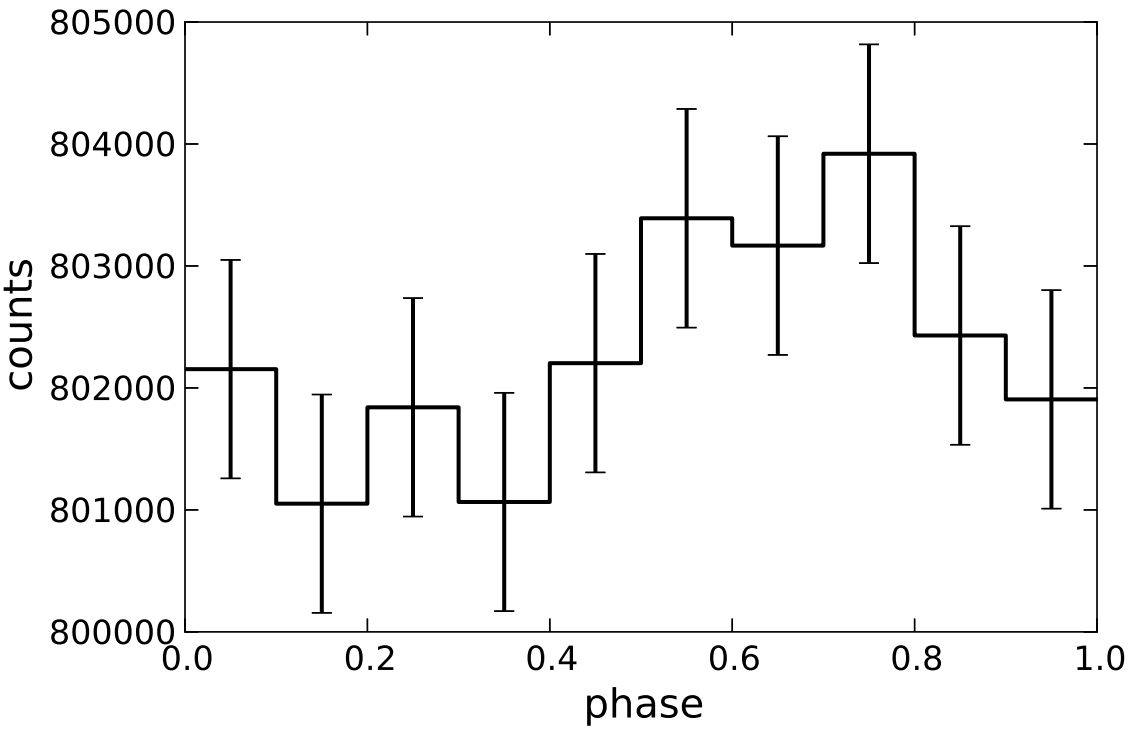

Figure 8.2: The observed lightcurve of J0337 as a function of phase in the band 4000-5500 Å, after folding by the 2.7 ms pulse period. The error bars for each bin are the square root of the total number of counts in the bin. The H test shows no statistically significant detection of periodicity.

5500 Å is shown in Figure 8.2. The error bars are the square root of the total counts in the bin. We applied a χ^2 test to this light curve to test for deviation from a flat line with a value set to the average count rate. The flat line would be the expected light curve if there is no pulsed emission. The result of the test was a χ^2 value of 10.0 with 9 degrees of freedom, corresponding to a false positive probability (FPP) of 35%. This indicates that the significance of pulsed emission, for the choice of binning used here, is only 0.94 σ.

For a bin-free test for periodicity we applied the H-test to the folded photon timestamps. The included photons numbered about eight million. The H test returned an H metric of 6.04, which corresponds to an FPP of 9.1% (or equivalently a 1.7 σ significance). This is higher than the binned test, but it still shows no statistically significant detection of pulsed emission.

The H test provides the number of Fourier modes that maximizes the H metric, so that a Fourier model of a periodic signal can be constructed with these modes. For our light curve, the H test indicated that the first mode (i.e. a sinusoid at the fundamental) was the most significant. We used simulated photon lists with pulse profiles of the form of our Fourier model to estimate a 95% confidence upper limit on the pulsed optical emission. We found that in order to have a 95% probability of observing a light curve with an H metric of 6.04 or higher, the pulsed emission (averaged over phase) would have to be at least 6.8 magnitudes fainter than the inner white dwarf companion. Since SDSS measures the g magnitude of the system to be 17.93 ± 0.01 (Alam et al., 2015), this sets the 95% confidence upper limit on the brightness of the pulsed emission at $g \sim 25\,\mathrm{mag}$.

8.4 Discussion

Although we found no statistically significant periodicity, the lightcurve in Fig 8.2 may show a hint of periodicity that could be uncovered with more observing time. It would be worthwhile to observe this object again with ARCONS or with one of its MKID successors that are currently in development (Meeker et al. 2016, Proc. AO4ELT, submitted).

Until there is an example of detected optical pulses in an MSP, we have no firm basis to predict how bright pulses would be. One generally useful value to predict the brightness of a pulsar is the observability metric, $\frac{\dot{E}}{d^2}$, where $\dot{E}$ is the rate of change in rotational energy and d is the distance to the pulsar. This metric indicates the energy available for emission in all wavelengths from a pulsar's slow down, with an assumption of isotropic emission. Thompson (2000) showed that the pulsars known at the time with the highest observability metric were the ones with known high energy emission (X-ray and γ-rays). For J0337, $\frac{\dot{E}}{d^2} = 2.0 \times 10^{34} ergs\ s^{-1} kpc^{-2}$ (Ransom et al., 2014). This places J0337 among the pulsars with higher values, of which many are detected in X-rays (including J0337; Spiewak et al. 2016, ApJ, submitted).

Shearer & Golden (2001) showed that the best predictor of optical brightness for the known optical pulsars is the strength of the B field at the pulsar's light cylinder, as formulated by Pacini (1971), with revisions in Pacini & Salvati (1983) and Pacini & Salvati (1987). Shearer & Golden (2001) did a regression fit to find that for the five pulsars with detected optical pulses, the optical luminosity scales roughly with $\dot{E}^{1.6}$. If this scaling holds true for MSPs, then for J0337 we would expect $m_V \sim 31$ mag. If this prediction is right, we will need a 30-m class telescope to detect optical emission.

If optical pulsation could eventually be found in an MSP with enough SNR, optical observations could be used for pulsar timing. Zampieri et al. (2014) demonstrate how to compute a timing solution for the Crab pulsar using optical timing data. Optical timing

would complement ongoing radio pulsar timing arrays, because it does not suffer from interstellar dispersion or scintillation and it may be able to help characterize the sources of noise in radio timing.

Chapter 9

Conclusions

Utilizing the unique abilities of MKIDs, we have shown that the enhancement of optical pulses in the Crab pulsar accompanied by a main pulse GRP has a significant dependence on the arrival phase of the GRP. Our data also shows some correlation between enhancement of optical main pulses and interpulse GRPs. In addition, the spectra of enhanced optical pulses and normal pulses do not differ significantly, and no significant relationship is detected between optical enhancement and GRP flux. Future observations with MKID instruments should improve the signal-to-noise ratio in probing the arrival phase to flux relationship in order to better resolve the apparent peak.

We have also presented a search for optical pulsations in the millisecond pulsar J0337+1715 at the pulse period seen in radio observations. No significant pulsations were found in the ARCONS passband 4000-11000 Å. In the range 4000-5500 Å, we have established a 95% confidence upper limit for phase-averaged pulsed emission at 6.8 magnitudes fainter than the inner white dwarf, which sets the limit at $g \sim 25\,\mathrm{mag}$.

Bibliography

Alam, S., et al. 2015, ApJS, 219, 12

Alpar, M. A., Cheng, A. F., Ruderman, M. A., & Shaham, J. 1982, Nature, 300, 728

Alpert, B. K., Horansky, R. D., Bennett, D. A., Doriese, W. B., Fowler, J. W., Hoover, A. S., Rabin, M. W., & Ullom, J. N. 2013, Review of Scientific Instruments, 84

Argyle, E., & Gower, J. F. R. 1972, ApJL, 175, L89

Barnard, J. J., & Arons, J. 1986, ApJ, 302, 138

Beskin, V. S., & Philippov, A. A. 2012, Monthly Notices of the Royal Astronomical Society, 425, 814

Bhattacharya, D., & van den Heuvel, E. P. J. 1991, Phys. Rep., 203, 1

Bietenholz, M. F., Kassim, N., Frail, D. A., Perley, R. A., Erickson, W. C., & Hajian, A. R. 1997, ApJ, 490, 291

Bilous, A. V., Kondratiev, V. I., McLaughlin, M. A., Ransom, S. M., Lyutikov, M., Mickaliger, M., & Langston, G. I. 2011, ApJ, 728, 110

Bottom, M. 2016, PhD thesis, California Institute of Technology

Bottom, M., et al. 2016, PASP, 128, 075003

Bourrion, O., Vescovi, C., Bouly, J. L., Benoit, A., Calvo, M., Gallin-Martel, L., Macias-Perez, J. F., & Monfardini, A. 2012, Journal of Instrumentation, 7, 7014

Chamberlin, H. 1980, Musical Applications of Microprocessors, The Hayden microcomputer series (Hayden Book Company)

Collins, S., Shearer, A., Stappers, B., Barbieri, C., Naletto, G., Zampieri, L., Verroi, E., & Gradari, S. 2011, Proceedings of the International Astronomical Union, 7, 296

Day, P. K., LeDuc, H. G., Mazin, B. A., Vayonakis, A., & Zmuidzinas, J. 2003, Nature, 425, 817

de Jager, O. C., Raubenheimer, B. C., & Swanepoel, J. W. H. 1986, A&A, 170, 187

Dekany, R., et al. 2013, ApJ, 776, 130

Doi, M., et al. 2010, AJ, 139, 1628

Duan, R. 2015, PhD thesis, California Institute of Technology

DuPlain, R., Benson, J., & Sessoms, E. 2008, in , 70191A–70191A–10

Enss, C. 2002, AIP Conference Proceedings, 605, 5

Fressin, F., et al. 2011, ApJS, 197, 5

Gao, J. 2008, PhD thesis, California Institute of Technology

Hankins, T. H., Kern, J. S., Weatherall, J. C., & Eilek, J. A. 2003, Nature, 422, 141

Harding, A. K., Stern, J. V., Dyks, J., & Frackowiak, M. 2008, ApJ, 680, 1378

Heiles, C., Campbell, D. B., & Rankin, J. M. 1970, Nature, 226, 529

Hobbs, G. B., Edwards, R. T., & Manchester, R. N. 2006, MNRAS, 369, 655

Hotan, A. W., van Straten, W., & Manchester, R. N. 2004, PASA, 21, 302

Hubmayr, J., et al. 2015, Applied Physics Letters, 106, 073505

Irwin, K. D., Hilton, G. C., Wollman, D. A., & Martinis, J. M. 1996, Applied Physics Letters, 69, 1945

Janssen, R. M. J., Baselmans, J. J. A., Endo, A., Ferrari, L., Yates, S. J. C., Baryshev, A. M., & Klapwijk, T. M. 2013, Applied Physics Letters, 103, 203503

Jessner, A., et al. 2010, A&A, 524, A60

Jovanovic, N., et al. 2015, PASP, 127, 890

Kaplan, D. L., van Kerkwijk, M. H., Koester, D., Stairs, I. H., Ransom, S. M., Archibald, A. M., Hessels, J. W. T., & Boyles, J. 2014, ApJ, 783, L23

Kargaltsev, O., Pavlov, G. G., & Romani, R. W. 2004, ApJ, 602, 327

Koptsevich, A. B., Lundqvist, P., Serafimovich, N. I., Shibanov, Y. A., & Sollerman, J. 2003, A&A, 400, 265

Kraus, H., von Feilitzsch, F., Jochum, J., Mssbauer, R., Peterreins, T., & Prbst, F. 1989, Physics Letters B, 231, 195

Kuzmin, A. D., Kondrat'ev, V. I., Kostyuk, S. V., Losovsky, B. Y., Popov, M. V., Soglasnov, V. A., D'Amico, N., & Montebugnoli, S. 2002, Astronomy Letters, 28, 251

Lommen, A., et al. 2007, ApJ, 657, 436

Lorimer, D. R. 2005, Living Reviews in Relativity, 8

Lundgren, S. C., Cordes, J. M., Ulmer, M., Matz, S. M., Lomatch, S., Foster, R. S., & Hankins, T. 1995, ApJ, 453, 433

Lyne, A., & Graham-Smith, F. 2012, Pulsar Astronomy, Cambridge Astrophysics (Cambridge University Press)

Lyne, A. G., Pritchard, R. S., & Graham-Smith, F. 1993, MNRAS, 265, 1003

Lyons, R. G. 2004, Understanding Digital Signal Processing (2nd Edition) (Upper Saddle River, NJ, USA: Prentice Hall PTR)

Lyutikov, M., & Parikh, A. 2000, ApJ, 541, 1016

Manchester, R. N., Peterson, B. A., & Wallace, P. T. 1984, Nature, 310, 569

Martinache, F., et al. 2014, PASP, 126, 565

Mazin, B. 2005, PhD thesis, California Institute of Technology

Mazin, B. A., et al. 2013, PASP, 125, 1348

McHugh, S., Mazin, B. A., Serfass, B., Meeker, S., O'Brien, K., Duan, R., Raffanti, R., & Werthimer, D. 2012, Review of Scientific Instruments, 83, 044702

Meeker, S., Mazin, B., Jensen-Clem, R., Walter, A., Szypryt, P., Strader, M., & Bockstiegel, C. 2015, Adaptive Optics for Extremely Large Telescopes 4 –Conference Proceedings, 1

Miceli, A., Cecil, T. W., Gades, L., & Quaranta, O. 2014, Journal of Low Temperature Physics, 176, 497

Mickaliger, M. B., et al. 2012, ApJ, 760, 64

Middleditch, J. H., Imamura, J. N., & Steiman-Cameron, T. Y. 1988, ApJ, 335, 753

Mignani, R. P., Manchester, R. N., & Pavlov, G. G. 2003, ApJ, 582, 978

Mignani, R. P., Pavlov, G. G., de Luca, A., & Caraveo, P. A. 2004, Advances in Space Research, 33, 518

Mignani, R. P., Pavlov, G. G., & Kargaltsev, O. 2011, A&A, 531, A105

Mutel, R. L., Broderick, J. J., Carr, T. D., Lynch, M., Desch, M., Warnock, W. W., & Klemperer, W. K. 1974, ApJ, 193, 279

Oosterbroek, T., et al. 2008, A&A, 488, 271

Pacini, F. 1971, ApJ, 163, L17

Pacini, F., & Salvati, M. 1983, ApJ, 274, 369

—. 1987, ApJ, 321, 447

Parsons, A., et al. 2006, in Asilomar Conference on Signals and Systems, Pacific Grove, CA, 2031–2035

Peacock, A., et al. 1997, Journal of Applied Physics, 81, 7641

Petrova, S. A. 2009, MNRAS, 395, 1723

Ransom, S. 2011, PRESTO: PulsaR Exploration and Search TOolkit, Astrophysics Source Code Library

Ransom, S. M., et al. 2014, Nature, 505, 520

Shearer, A., & Golden, A. 2001, ApJ, 547, 967

Shearer, A., Stappers, B., O'Connor, P., Golden, A., Strom, R., Redfern, M., & Ryan, O. 2003, Science, 301, 493

Shklovsky, I. S. 1970, ApJL, 159, L77

Siverd, R. J., et al. 2012, ApJ, 761, 123

Słowikowska, A., Kanbach, G., Kramer, M., & Stefanescu, A. 2009, MNRAS, 397, 103

Staelin, D. H., & Sutton, J. M. 1970, Nature, 226, 69

Strader, M. J., et al. 2013, ApJ, 779, L12

—. 2016, MNRAS, 459, 427

Sutaria, F. K., Ray, A., Reisenegger, A., Hertling, G., Quintana, H., & Minniti, D. 2003, A&A, 406, 245

Swenson, L. J., et al. 2012, in Proc. SPIE, Vol. 8452, Millimeter, Submillimeter, and Far-Infrared Detectors and Instrumentation for Astronomy VI, 84520P

Szypryt, P., et al. 2014, MNRAS, 439, 2765

Thompson, D. J. 2000, Advances in Space Research, 25, 659

Ulbricht, G., Mazin, B. A., Szypryt, P., Walter, A. B., Bockstiegel, C., & Bumble, B. 2015, Applied Physics Letters, 106, 251103

van Eyken, J. C., et al. 2015, ApJS, 219, 14

van Rantwijk, J., Grim, M., van Loon, D., Yates, S., Baryshev, A., & Baselmans, J. 2016, IEEE Transactions on Microwave Theory and Techniques 2016, 64, 1876

VERITAS Collaboration et al. 2011, Science, 334, 69

Yates, S. J. C., Baselmans, J. J. A., Endo, A., Janssen, R. M. J., Ferrari, L., Diener, P., & Baryshev, A. M. 2011, Applied Physics Letters, 99, 073505

Zampieri, L., et al. 2014, MNRAS, 439, 2813