UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

DESIGN AND IMPLEMENTATION OF AN ALL-COTS DIGITAL BACK-END FOR A PULSE-DOPPLER SYNTHETIC APERTURE RADAR

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE

By

RUSSELL KENNEY
Norman, Oklahoma

2020

DESIGN AND IMPLEMENTATION OF AN ALL-COTS DIGITAL BACK-END
FOR A PULSE-DOPPLER SYNTHETIC APERTURE RADAR

A THESIS APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY THE COMMITTEE CONSISTING OF

Dr. Jay McDaniel, Chair

Dr. Hjalti Sigmarsson

Dr. Mark Yeary

Acknowledgments

I want to thank my advisors, Dr. Jay McDaniel, Dr. Hjalti Sigmarsson, and Dr. Mark Yeary for their support on this project. I have learned an enormous amount from them, not only about RF engineering and radar but also about professional development, academia, and devotion to one's work. I am also thankful that on top of being my mentors, I can also call them my friends. I want to thank Dr. McDaniel as well for pushing me to be my best and to pursue education at the highest level, and for encouraging me in my research when I doubted my own abilities.

I also want to thank my brother, Dwaine Kenney, and my parents, Dwaine and Diane Kenney. I have never once doubted their support and I'm so grateful that they have backed me up in everything, celebrating with me in victory and encouraging me in defeat.

Finally, I want to thank my wonderful fianceé, Sarah Crooks. She has encouraged me more than anyone to pursue my goals in education, and has made great sacrifices to help me balance our time together with my time spent on research. She is an excellent partner in my work and in my faith, and I am extremely blessed to spend the rest of my life with her.

This work was funded by the Department of Energy's Kansas City National Security Campus, operated by Honeywell Federal Manufacturing and Technologies, LLC, under contract number DE-NA0002839.

Table of Contents

List of Tables

List of Figures

Abstract

Radar imaging techniques employing synthetic aperture radar (SAR) are ubiquitous in applications such as defense, remote sensing, space exploration, terrain mapping, and many others. However, to obtain fine image resolution, radar systems must be capable of utilizing large signal bandwidths. By the sampling theorem, a large signal bandwidth equates to a high sampling frequency, resulting in more expensive and complex digital electronics required to digitize and process the waveform. Using linear frequency modulated (LFM) pulses and stretch processing techniques, systems such as frequency-modulated continuous-wave (FMCW) radars reduce the required sampling rate at the expense of longer pulses, higher transmit duty cycle, and decreased pulse repetition frequency. While these tradeoffs are often acceptable, in many situations they are not, and a pulse-Doppler radar system is required. These systems can utilize LFM pulses with nearly any desired pulse length and pulse repetition frequency to perform imaging, but they must have an analog-to-digital converter (ADC) and back-end processing capable of handling the full waveform bandwidth, leading to increased cost, size, or both.

At the University of Oklahoma's Advanced Radar Research Center, a pulse-Doppler radar system for use in a SAR application is designed and built using only commercially available components to decrease the size and cost of the radar, specifically the digital back-end. A minimum size and weight is targeted for this system because it is desired to eventually fly the radar and form images on a lightweight airborne platform, such as a quad- or octo-copter. The challenge with using commercial parts for a custom digital pulse-Doppler radar is that it is difficult to meet the strict timing requirements inherent to pulse-Doppler radar while simultaneously meeting the high-bandwidth requirements imposed by SAR. In this thesis, the design and implementation of the digital back-end for the custom SAR system is presented. The focus is placed on designing a control system and clock distribution scheme in the digital back-end to ensure pulse to pulse coherence while maintaining ideal LFM spectral quality. Additionally, a calibration method is devised to provide accurate range measurements each time the radar is turned on even if the latency between the digital transmitter and receiver changes. At the conclusion of this work, it is shown that the radar system is capable of performing accurate pulse-Doppler radar through the generation of range-Doppler maps from data captured by the radar. The results of these tests indicate that the system is suitable for eventual use in SAR imaging applications.

Chapter 1

Introduction

Radar has been used for a number of applications in modern life, including meteorology, national defense, terrain mapping, and air traffic control, to name a few. In all of these applications, the radar system is often designed to be a pulse-Doppler system, which typically increases the complexity of the digital electronics required when compared to continuous-wave (CW) and frequency-modulated continuous-wave (FMCW) radar systems [3]. This added complexity often leads to added size, weight, power, and cost (SWaP-C) of the system. For example, some radar systems use a pre-made digital chassis, such as an NI-DAQ, to synthesize the transmitted waveform and to digitize and process the received waveform [4]. While such a system conveniently abstracts away the design of the digital transceiver, they are also typically large, heavy, and prohibitively expensive. Although the increased SWaP-C is tolerable for ground-based radar platforms, these systems are not easily reproducible for bulk-production and cannot be placed on lightweight aerial platforms such as small UAVs or quad/octo-copters.

One solution to this problem is to use small and cheap commercial-off-the-shelf (COTS) digital parts to construct the digital back-end of the radar. The use of COTS parts allows for a low-cost and easily replicated digital system that can be integrated into a small form-factor printed circuit board (PCB) that satisfies the low SWaP-C requirement. The primary challenge of using COTS parts to build a digital radar is that typically the digital parts are not designed to specifically meet the strict timing requirements of a pulse-Doppler radar system. If care is not taken in the design and integration of the digital parts, particularly with regard to clock synchronization between the devices, the radar system will not operate as intended due to digital timing errors, which lead to ambiguities in the range measurements gathered by the radar. To further complicate the design, larger analog bandwidths and shorter pulse widths are desirable to give a more precise range resolution and a smaller blind range [5], which requires an increase in the sampling frequency of the analog-to-digital converter (ADC) and the reconstruction frequency of the digital-to-analog converter (DAC). As these frequencies increase, it becomes more difficult to synchronize the digital clocks operating on each device.

This thesis is concerned with the digital design of a high-sampling frequency wideband pulse-Doppler radar system for use in a synthetic aperture radar (SAR) airborne imaging application. The system is desired to be small and light enough that it can be placed on a light octo-copter, and must be robust enough to obtain a range resolution of 1 m with a look angle of 30°. Although some low SWaP-C SAR systems have been developed in industry and the literature [6],[7], these are typically not small and light enough to be placed on a small drone, and the systems that are small enough usually simplify their hardware through FMCW processing techniques. The system designed in this work is aimed to improve upon the current state-of-the-art by enabling pulse-Doppler SAR operation onboard a small, cheap, and lightweight air platform, such as a quad- or octo-copter.

1.1 Thesis Outline

This thesis is divided into the following chapters. In Chapter 2, the background of digital pulse-Doppler radar is covered. This includes the mathematical basis for pulse-Doppler radar processing techniques and the relationships between the processing parameters and the radar characteristics. This chapter also describes the basics of digital radar architectures with a focus on the digital synthesis and capture of an analog radio frequency (RF) waveform.

In Chapter 3, the initial design of the proposed radar system is described. An overview of the radar characteristics including center frequency, bandwidth, and pulse rate is given here. The hardware used in the radar is also described with a focus on the selection of the hardware comprising the digital back-end.

In Chapter 4, the details of the implementation of the digital back-end are discussed. This chapter focuses on low-level details of configuring the digital hardware and includes some additions to the original system design to fix problems with the waveform spectral integrity. This chapter concludes with a demonstration of the radar pulse integrity when it is transmitted in a full loopback from the digital back-end, to the RF front-end, and back to the digital back-end.

In Chapter 5, issues related to the coherence of the radar from pulse to pulse are discussed and resolved. The chapter describes the sources of digital incoherence and explains the method by which the incoherence is eliminated. Modifications to the digital back-end and RF front-end are proposed and implemented to allow pulse-to-pulse coherence without sacrificing the spectral integrity of the waveform.

In Chapter 6, a pulse timing problem causing inaccurate range measurements when the radar is restarted is discussed. A loopback calibration solution is proposed, and a derivation of an algorithm for calibrating the radar on each system reset is presented. The additional hardware used to implement this calibration is also presented and results showing its proper operation are given.

In Chapter 7, the completed radar system is presented, and final improvements to enhance the functionality and usability of the radar are proposed. An overview of each proposed improvement is given. Finally, a full system test of the radar including range-Doppler measurements of moving targets is performed, verifying that the system is fully operable.

In Chapter 8, the conclusion to the thesis and propositions for future improvements and measurements are given.

Chapter 2

Background

To fully grasp the scope of the work demonstrated in this thesis, the basics of digital pulse-Doppler radar systems must be understood. This includes an understanding of the theory of operation of a pulse-Doppler radar, the processing steps, and the implementation details of various components in the digital back-end. As such, this section seeks to provide a foundational understanding of pulse-Doppler radar operation, the processing required to obtain useful information with a pulse-Doppler radar, and typical digital radar system operation and design considerations.

2.1 Fundamentals of Pulse-Doppler Radar

A pulse-Doppler radar system operates by transmitting a train of short pulses with a low duty cycle and then capturing the pulses as they return to the radar after reflecting off objects (*targets* or *scatterers*). The pulses have a pulse width of T_p and are transmitted at a precise frequency called the *pulse repetition frequency*, or PRF. The period of the PRF, or the time between successive pulses, is known as the *pulse repetition interval*, or PRI. Typically the pulse width is much shorter than the PRI. A short pulse train consisting of five pulses with a PRI of 400 μs and a pulse width of 50 μs of is shown in Figure 2.1.

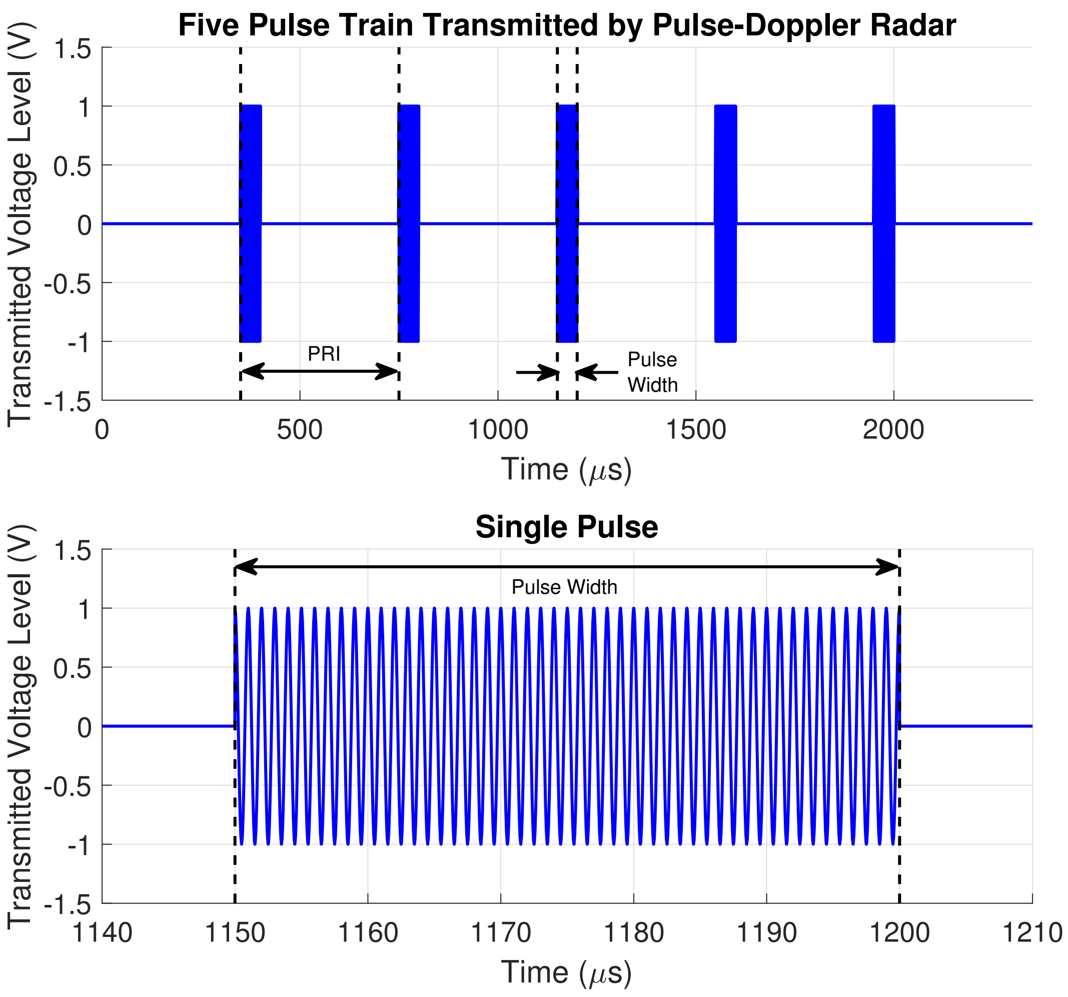


Figure 2.1: A simple pulse train

The radar is able to measure both the distance between the scatterers and the radar (R) and relative radial velocity of the scatterers with respect to the radar (v_s). The range R can be extracted from the time delay between the time that a pulse is transmitted and the time the pulse is received, denoted by τ_p. The relative radial velocity of the target can be extracted by determining the Doppler shift of the received waveform when compared to the original waveform, denoted by F_D. The range is calculated from τ_p by

$$R = \frac{c\tau_p}{2} \tag{2.1}$$

where c is the speed of light. This equation accounts for the two-way propagation time of the radar pulse from the radar to the scatterer and then back.

To determine the radial velocity of a moving scatterer, the frequency of the Doppler shift caused by the relative motion of the scatterer can be extracted over the course of multiple pulses called a *coherent processing interval* (CPI). The Doppler shift frequency (F_D) is calculated by

$$F_D = \frac{2v_s}{c - v_s} F_c \approx \frac{2v_s}{c} F_c = \frac{2v_s}{\lambda_c} \tag{2.2}$$

where v_s is the radial velocity of the scatterer and F_c and λ_c are the frequency and wavelength of the carrier, respectively [5]. The Doppler shift can be either positive or negative. By convention, a positive F_D occurs from the movement of a target toward the radar, while a negative F_D occurs from the movement of a target away from the radar. A discussion of how to extract F_D from the radar data is given in Section 2.1.1.

Given two scatterers within the observable scene of the radar, the range to each target cannot be resolved with infinite precision. Instead, the radar can only resolve targets into discrete "range bins," where the size of each range bin is denoted as the *range resolution* (ΔR) of the radar. The range resolution of the radar is given by

$$\Delta R = \frac{c}{2\beta} \tag{2.3}$$

where β is the instantaneous bandwidth of the transmitted pulse [5]. For the simplest pulse waveform, each pulse is a single frequency tone with a square pulse amplitude envelope, as seen in Figure 2.1. For this case, β is approximately equal to $\frac{1}{T_p}$ due to the Fourier uncertaintly principle. It is typical for a radar system to implement some sort of modulation to the pulse to increase the bandwidth without decreasing the pulse width. A frequently implemented pulse waveform is the *linear frequency modulated* (LFM) pulse. The LFM waveform, also commonly called a "chirp," increases linearly in frequency from f_1 to f_2 over the course of the pulse width, as shown in Figure 2.2. At complex baseband, f_1 is $-\frac{\beta}{2}$ and f_2 is $\frac{\beta}{2}$. The complex envelope of such a waveform is given by

$$x(t) = a(t) \exp\left[j2\pi\left(-\frac{\beta}{2}\right)t + j\pi\frac{\beta}{T_p}t^2\right] \quad 0 \leq t \leq T_p \tag{2.4}$$

where $a(t)$ is some amplitude scale factor introduced by the RF front-end and the bandwidth of the waveform is given by

$$\beta = f_2 - f_1 \tag{2.5}$$

which is independent of the pulse width. Thus, a radar can obtain an arbitrarily fine resolution by increasing the band of frequencies covered by each LFM pulse.

2.1.1 Pulse Compression and Doppler Processing

When a train of pulses is transmitted, the received signal may be completely obscured by noise. For targets a considerable distance from the radar, the received reflection power is potentially much lower than the thermal noise power introduced by the system itself, making the discovery of target reflections within the received signal difficult. The balance of received signal power compared to the noise power is referred to as the *signal-to-noise ratio* (SNR). In order to reliably extract reflection signals in the noise received signal, it is desirable to implement a digital filter as a processing step that maximizes the SNR of the received signal.

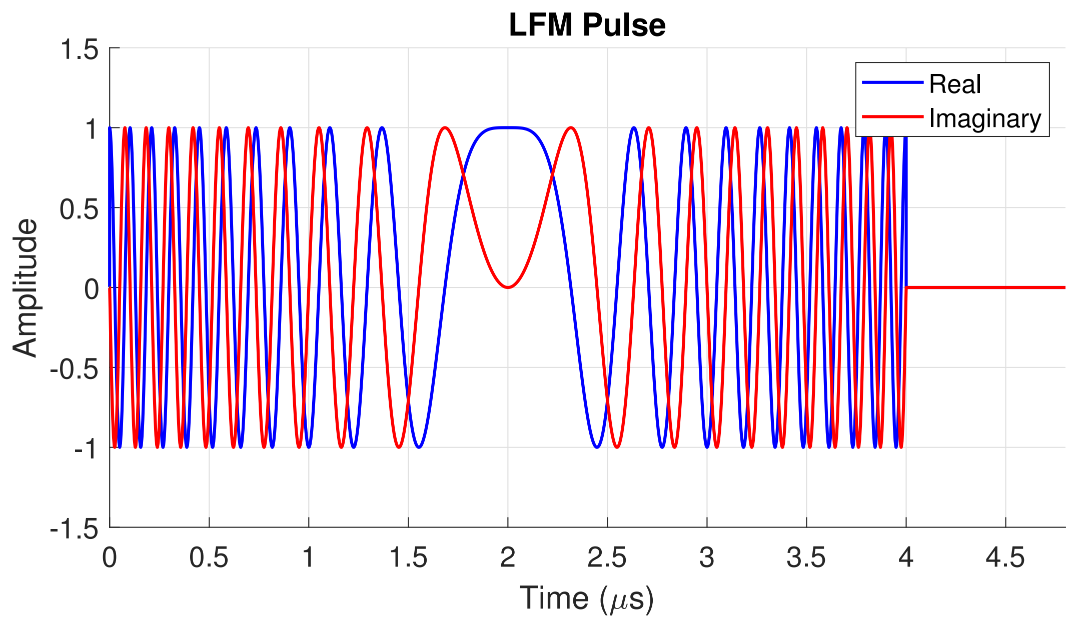


Figure 2.2: A simple complex LFM pulse

The filter response $h(t)$ that maximizes the SNR is known as the *matched filter*. It is called the matched filter because the impulse response is matched to the signal being transmitted by the radar. Say that a radar transmits an LFM pulse whose complex envelope is given by (2.4). The matched filter $h(t)$ used to maximize the SNR of the received signal will be the time-reversed, complex conjugate of the signal. That is,

$$\begin{aligned} h(t) &= x^*(T_p - t) \\ &= a^*(T_p - t)\exp\left[j2\pi\left(\frac{\beta}{2}\right)(T_p - t) - j\pi\frac{\beta}{T_p}(T_p - t)^2\right] \quad 0 \le t \le T_p \ . \end{aligned} \tag{2.6}$$

The inclusion of T_p in the argument above ensures that the matched filter is causal. The convolution of some received signal $r(t)$ with $h(t)$ is given by

$$y(t) = \int_{-\infty}^{\infty} r(\tau)h^*(t-\tau)d\tau \tag{2.7}$$

which can be rearranged as

$$\begin{aligned} y(t) &= \int_{-\infty}^{\infty} r(\tau)h^*(t-\tau)d\tau \\ &= \int_{-\infty}^{\infty} r(\tau)x^*(\tau+T_p-t)d\tau \\ &= \int_{-\infty}^{\infty} x^*(\tau)r(\tau+t-T_p)d\tau \\ &= R_{xr}(t-T_p) \end{aligned} \tag{2.8}$$

where the resulting $y(t)$ is referred to as the *range profile* of the received data $r(t)$ and $R_{rx}(t-T_p)$ is the cross-correlation of the received signal $r(t)$ with the ideal pulse waveform $x(t)$ shifted in time by the pulse width T_p. The result of (2.8) shows that the matched filter operation is equivalent to the mathematical cross-correlation between the received signal and ideal pulse. The range profile $r(t)$ indicates the magnitude and phase of a scatterer return at any time t, which can be correlated to range R by using (2.1).

The matched filter works by compressing the energy of each pulse into a smaller span of time so that a "peak" will appear, indicating the presence of the desired echo waveform pattern at a particular time. For this reason, matched filtering is sometimes referred to as *pulse compression*. The peak appears at time $\tau_p + T_p$, which is the sum of the propagation time of the pulse and the delay of the causal matched filter. This peak has a 3-dB width approximately equal to $\frac{1}{\beta}$ in units of time, which can be converted to range using (2.1). The resulting expression is equivalent to the range resolution of the radar given in (2.3). The matched filter output of an LFM pulse with a bandwidth of 20 MHz is shown in Figure 2.3.

As stated above, the matched filter allows a signal to be extracted from a very noisy data capture. Figure 2.4 shows the matched filter output of a received LFM

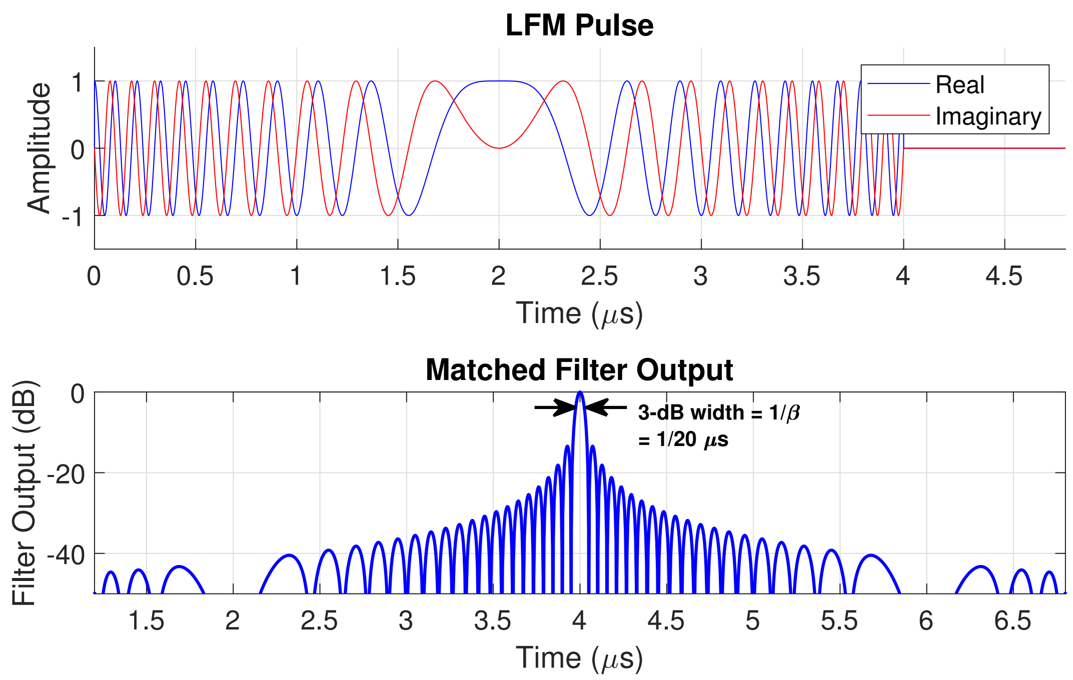


Figure 2.3: Matched filter output

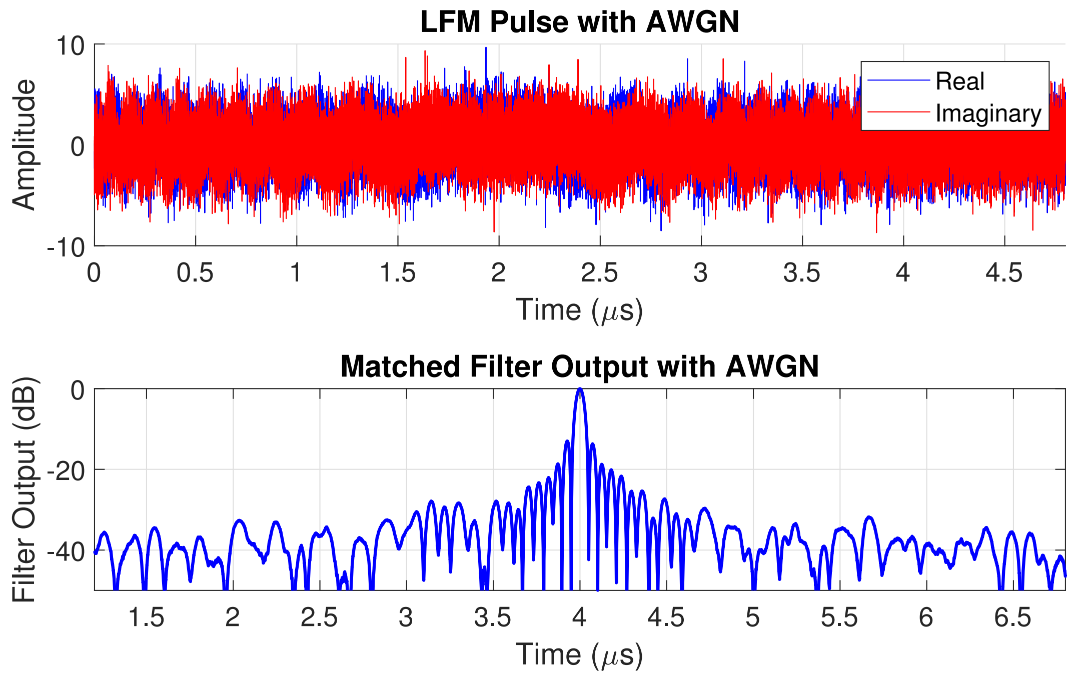


Figure 2.4: Matched filter output with noise

signal corrupted with additive white Gaussian noise (AWGN) when the the noise power of the un-processed signal is higher than the signal power.

When a pulse train similar to the one shown in Figure 2.1 is transmitted, the resulting data can be organized into two dimensions: one dimension in terms of ADC samples for each pulse (called “fast-time”) and the other dimension in terms of pulses (called “slow-time”). The pulse compression operation, performed in the fast-time dimension, will generate peaks at the target ranges for each pulse. If the target is not moving or is moving slowly, or if the CPI is very short, then the pulse compression peaks will occur at the same range for each pulse. An image of the pulse-compressed reflection intensity as a function of range and pulse number yields a “range-pulse map” of the radar scene. An example of a range-pulse map of a target moving toward the radar at 5 m/s at a range of 100 m is shown in Figure 2.5.

On top of sensing target range, pulse-Doppler radar systems are also useful for sensing target velocity. The primary vehicle for measuring velocity is the Doppler shift frequency of the received waveform that is caused by the relative radial ve-

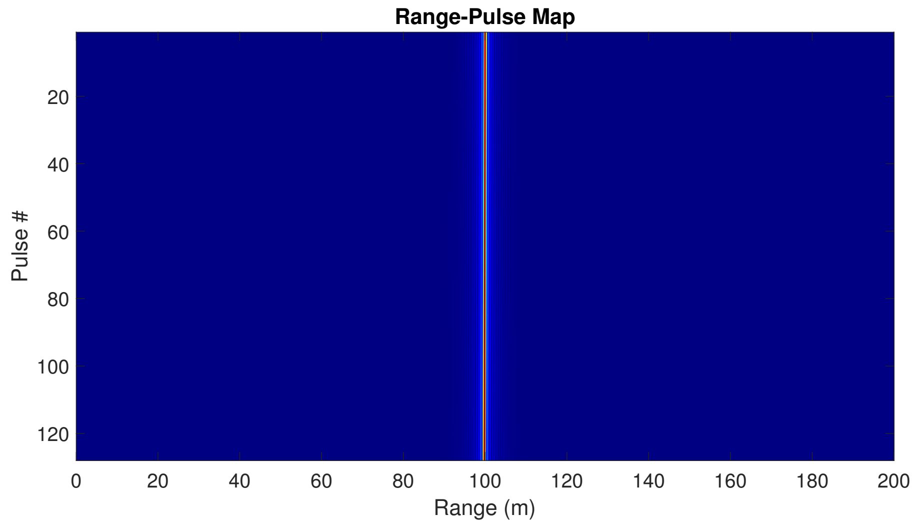


Figure 2.5: A simple range-pulse map. The range axis corresponds to “fast-time” and the pulse axis corresponds to “slow-time”

locity between the radar and the target. Generally, the Doppler shift frequency is on the order of kHz. Because pulses are on the order of microseconds long, the Doppler shift will typically not cause a substantial phase rotation of the received waveform over the course of a single pulse, meaning that it is not possible to extract the Doppler shift from a single pulse. To observe the Doppler shift over a sufficiently long time frame, a pulse train is transmitted to effectively "sample" the Doppler frequency at the PRF. Therefore, each pulse can be thought of as a digital sample of the Doppler shift. It follows then, that the Doppler shift can be extracted from the received pulse train by taking the Fourier Transform in the slow-time dimension. Taking the fast Fourier Transform (FFT) in the slow-time dimension at every range bin yields a peak located at the range and Doppler shift frequency of scatterers in the scene. Solving (2.2) for v_s shows that the Doppler shift frequency can be directly correlated to velocity. The 2D intensity image of return energy as a function of range and velocity (or Doppler shift) is called a "range-Doppler map." The range-Doppler map generated by performing Doppler processing on the range-pulse map is shown in Figure 2.6. Notice the single bright dot located at 100 m in range and 5 m/s, indicating the presence of a moving scatterer.

Another consideration when performing range and Doppler processing with a pulse-Doppler radar is that there is a tradeoff between maximum possible range measurement and maximum possible velocity measurement. These are referred to as the *maximum unambiguous range* and *maximum unambigous velocity*, and they are both related to the PRF of the radar. Consider a scatterer whose range to the radar R is large enough such that τ_p is greater than the PRI of the radar. The return from the target due to the first pulse will be captured after the second pulse has been transmitted. If this is the case, there is no way to tell whether the target return is due to the first or second pulse. Therefore, for a target to be unambiguously measured in

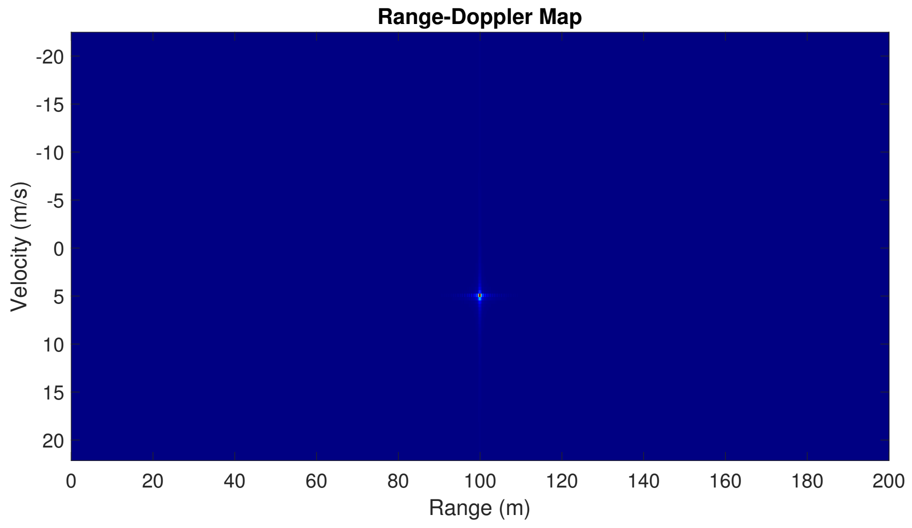


Figure 2.6: A simple range-Doppler map

range, its propagation delay τ_p must be less than the PRI. Therefore, the maximum unambiguous range (R_{max}) is given by

$$R_{\text{max}} = \frac{c \times PRI}{2} = \frac{c}{2 \times PRF} \tag{2.9}$$

Consider also that the Doppler shift frequency, from which target velocities are derived, is obtained by “sampling” the Doppler shift with each pulse at a “sampling frequency” equal to the PRF. Therefore, by the Nyquist Theorem, the maximum Doppler shift that can be measured unambiguously $F_{D,\text{max}}$ is given by

$$F_{D,\text{max}} = \frac{PRF}{2} \tag{2.10}$$

Note that positive and negative Doppler shifts can still be captured. Therefore, combining (2.10) and (2.2), the maximum unambiguous velocity v_{max} is given by

$$v_{\text{max}} = \frac{\lambda_c F_{D,\text{max}}}{2} = \frac{\lambda_c \times PRF}{4} \quad . \tag{2.11}$$

Note that the maximum unambiguous velocity is proportional to the PRF, while the maximum unambiguous range is inversely proportional to the PRF. Therefore, the maximum range and velocity that a pulse-Doppler radar can reliably measure are in conflict with one another. This is an important consideration in SAR processing, since the radar should be capable of measuring the Doppler shift caused by the moving ground below the flight platform while still unambiguously measuring the range to the ground, which is potentially a large distance.

2.1.2 Range Ambiguities Due to Digital Incoherence

Due to practical delays introduced by the system itself, the radar cannot measure the waveform propagation time τ_p directly. Instead, the actual delay measured by the radar (τ_m) is the sum of four primary components: the actual propagation time of the waveform from the radar to the target and back (τ_p), the delay of the waveform as it is conditioned by the RF transceiver (τ_{RF}), the matched filter delay T_p, and the delay introduced by the digital back-end as the waveform is synthesized and digitally captured (τ_{dig}). That is, τ_m is given by

$$\tau_m = \tau_p + \tau_{\text{RF}} + T_p + \tau_{\text{dig}} \quad . \tag{2.12}$$

The measured range before calibration will then be given using (2.1) with τ_m in place of τ_p. Ideally, both τ_{RF} and τ_{dig} are known constants and can be easily accounted for in processing. While this is typically true of τ_{RF}, τ_{dig} may be random in nature, introducing potentially large errors in the computation of R. This is especially problematic if τ_{dig} varies from pulse to pulse.

To demonstrate the negative impact that a random τ_{dig} can have on the range estimation given by a radar system, suppose that τ_{dig} is a random variable distributed uniformly in the interval $< -\tau_r, \tau_r >$. Assuming that τ_{RF} is known and that τ_{RF} and T_p are calibrated out of the equation, the transformation of τ_m to R by (2.1) gives

$$R_m = \frac{c\tau_p}{2} + \frac{c\tau_{\text{dig}}}{2} = R + R_{\text{dig}} \tag{2.13}$$

where R_m is the measured range, R is the actual range, and R_{dig} is the range error introduced by varying τ_{dig}. If τ_{dig} varies randomly from pulse to pulse, there is no way to compensate for it, and the range error of R_m will vary within the interval $< -\frac{c\tau_{\text{r}}}{2}, \frac{c\tau_{\text{r}}}{2} >$. If τ_r is even as large as 50 ns, the measured range for a stationary scatterer can vary by as much as 15 m, which is an unacceptable ambiguity for any radar with moderate range resolution. When the incoherent pulses are compressed to form a range-pulse map, as in Figure 2.7, the problem becomes evident, as the true range to the target clearly cannot be seen. After Doppler processing, the incoherence results in essentially a large amplitude noise signal, as shown in Figure 2.8.

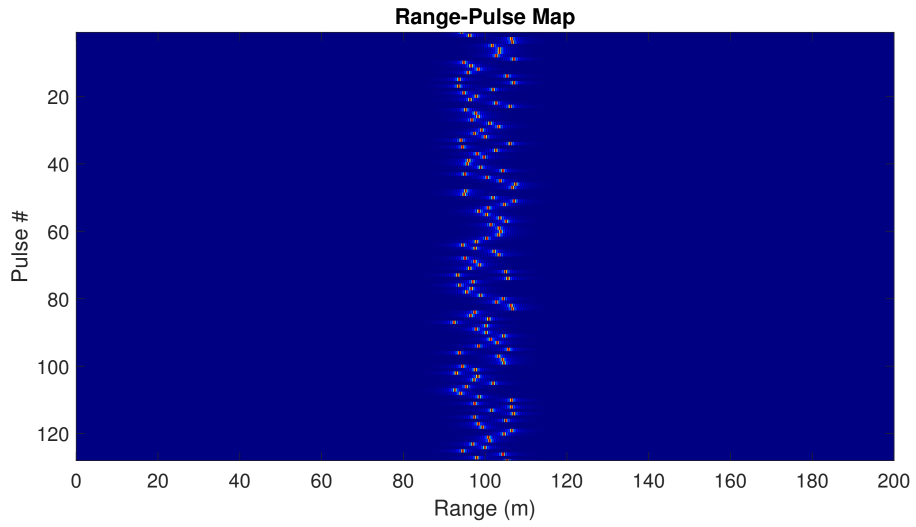


Figure 2.7: A range-pulse map with incoherent pulses

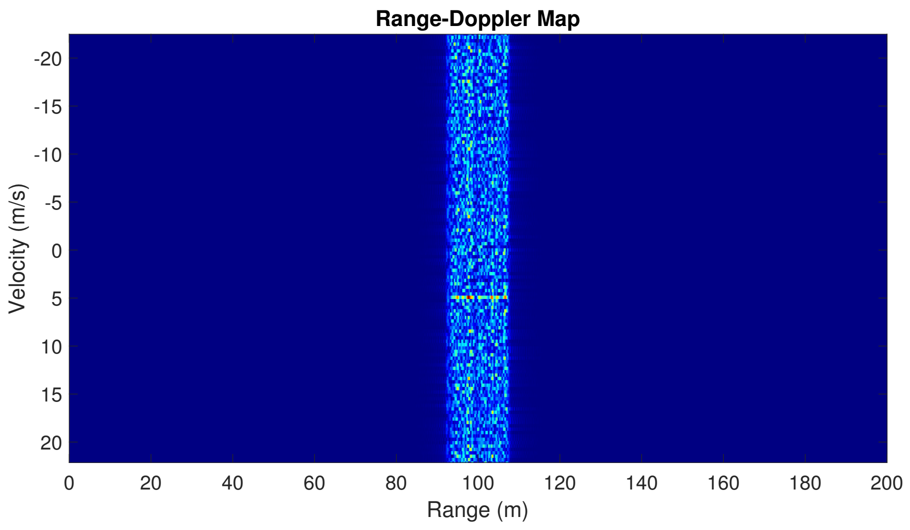


Figure 2.8: A range-Doppler map with incoherent pulses

2.2 Digital Radar Architecture

In a typical digital radar system, the radar can be broken up into two main sections: the RF front-end and the digital back-end. The digital back-end synthesizes a digital waveform, often as a complex baseband waveform. This digital waveform is then reconstructed as an analog waveform and fed to the RF front-end. The RF front-end conditions the signal for transmission, usually through analog filtering, amplification, and mixing with a high-frequency local oscillator (LO) to convert the waveform to the carrier frequency. The waveform is then coupled into free space by an antenna. When the waveform returns, an antenna captures the propagating wave and feeds it to the receive section of the RF front-end. The RF front-end conditions the signal for digital storage by amplification, removal of out-of-band interferers through analog filtering, and mixing back down to an intermediate frequency (IF) using the same LO as the transmitter to maintain coherence. The digital back-end digitizes the conditioned received waveform and stores it, and the digitized data is then processed to detect targets, extract range and velocity information, form images, or other processing steps. Depending on the application, this processing can be done in real-time or after the data has been collected in a post-processing step. The focus of this section is on the components and operation of a typical digital back-end.

2.2.1 Digital Waveform Synthesis

As alluded to above, the digital back-end of a radar has two primary roles: synthesizing the transmitted waveform and digitizing and processing the received waveform. To synthesize the waveform, the digital back-end must both digitally define the waveform to be transmitted and then convert the waveform from a digital sequence to an analog signal.

Before a waveform can be properly converted from digital to analog, it must be accurately defined as a digital signal. An increasingly common method for doing this is called *direct digital synthesis*, which is performed by a device called a *direct digital synthesizer* (DDS). A DDS enables the generation of complex waveforms without an external processor defining each point in the digital waveform sequence [8]. Instead, the DDS can synthesize waveforms with only preliminary programming through SPI and a sufficiently high frequency clock signal.

The basic DDS architecture is shown in Figure 2.9. The DDS synthesizes waveforms using a numerically controlled oscillator (NCO). The NCO consists of two components: a phase accumulator and a phase-to-amplitude converter. The phase accumulator is a register of N bits whose value corresponds to the current phase of the NCO such that each of the 2^N possible values are evenly spaced in the range $[0, 2\pi)$. The phase accumulator is incremented by some amount θ_{inc} every clock cycle of the DDS, where θ_{inc} is also defined by an N bit register mapping it to the range $[0, 2\pi)$. Given a DDS frequency of f_{DDS}, the output frequency of the DDS is

$$f_{\text{out}} = \frac{\theta_{\text{inc}}}{2\pi} f_{\text{DDS}} \quad . \tag{2.14}$$

Note that in order to satisfy the Nyquist Theorem, θ_{inc} should be less than π; otherwise, the DDS will synthesize an aliased frequency.

At every iteration of the DDS clock, the value of the phase accumulator is input to the phase-to-amplitude converter. The phase-to-amplitude converter determines the amplitude value of a sine or cosine wave for the given phase value, often using a lookup table of amplitude values.

The DDS conveniently allows for quick changes in output frequency simply by modifying the value of θ_{inc}. Some DDS products implement a digital ramp feature, which allows θ_{inc} to be swept upward or downward quickly, enabling the generation of the LFM waveform. Additionally, in some DDS products, the phase of the phase accumulator and the amplitude of the output can be modified, allowing for modulation of the amplitude and the phase of the output waveform in addition to the frequency.

The conversion from digital to analog requires the use of a digital-to-analog converter (DAC). Most DDS products include a DAC integrated within the same

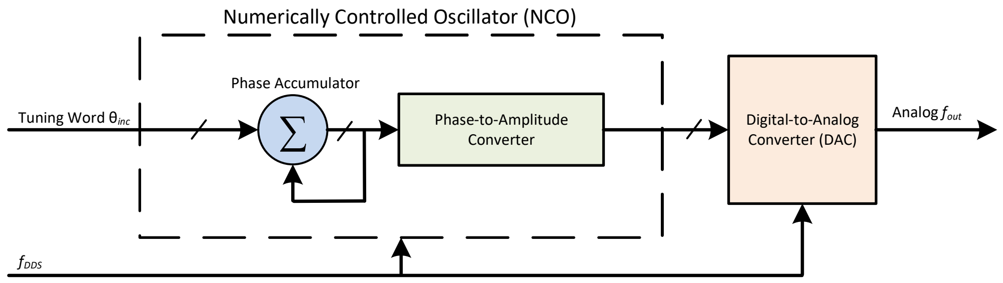

Numerically Controlled Oscillator (NCO)
Phase Accumulator
Tuning Word θ_{inc}
Σ
Phase-to-Amplitude Converter
Digital-to-Analog Converter (DAC)
Analog f_{out}
f_{DDS}

Figure 2.9: The basic archtecture of a DDS

integrated circuit (IC) containing the DDS core described above. As such, the DAC will often use the DDS clock to time the output of each sample. Each sample of the waveform output by the DAC does not transition smoothly into the next. Instead, each sample is held constant at its value for the entire sample period before transitioning to the next sample. The resulting output waveform has a stairstep appearance to it, resulting from the unfiltered harmonics of the digital signal. An analog filter, often referred to as a reconstruction filter, at the output of the DAC removes the harmonics and smooths the waveform into the desired sinusoid, as shown in Figure 2.10.

2.2.2 Digitization of Received Signal

In addition to synthesizing the transmit waveform, the radar digital back-end must also convert the received signal back into a digital signal to be stored and processed. Nominally, this step requires only an analog-to-digital converter (ADC)

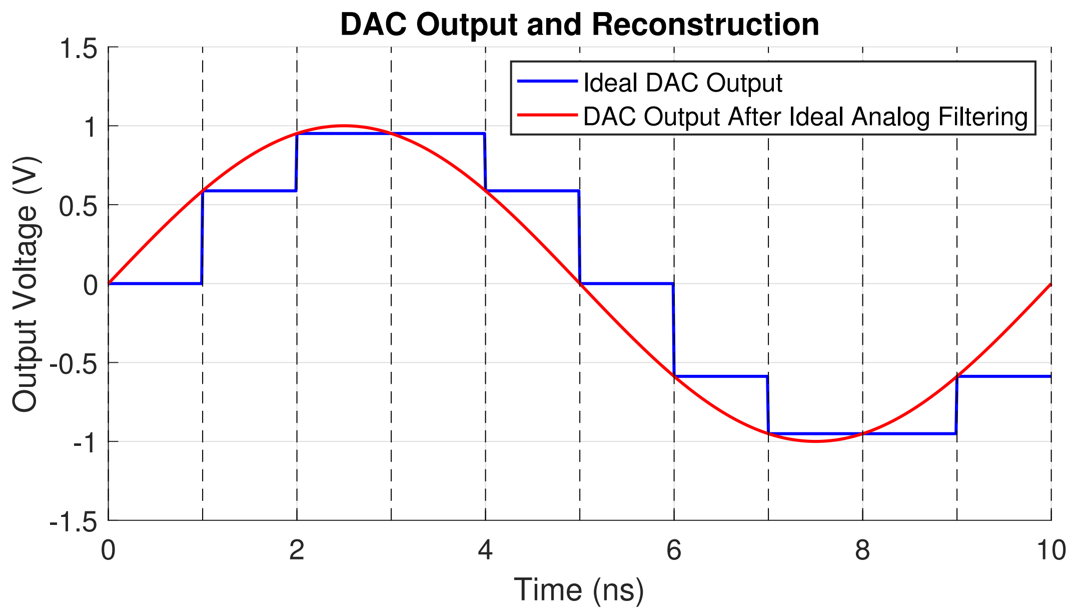


Figure 2.10: The ideal DAC output before and after analog filtering

with a sampling frequency f_s at least two times greater than the bandwidth β of the radar signal. However, if the sampling frequency and bit resolution of the ADC are high enough, the ADC will generate a massive amount of data every second that must be reliably stored in memory. For instance, given a signal bandwidth β of 100 MHz sampled by a 10-bit ADC at a sampling frequency of 200 MHz, the data channel communicating each sample of the ADC into memory must be capable of handling 2 Gbps at a minimum. To accommodate larger signal bandwidths and better receiver dynamic range, the number of bits per sample and the sampling frequency can become very large, which makes the high-speed capture of the ADC data difficult.

In order to handle these high data rates, radar systems often connect the ADC to a field-programmable gate array (FPGA) with some high speed data transfer protocol. The FPGA is then configured by the user to process and store the data, or transfer it to a computer for processing. Typically, the ADC will begin sampling as soon as it is turned on, so the FPGA must be programmed to decide when to store the samples generated by the ADC in memory. This may be accomplished by applying some internal or external trigger command to the FPGA to signal that data should be captured from the ADC. The trigger command issued to the FPGA often must be issued synchronously with the clock of the high-speed communication protocol. Therefore, the configuration of the high-speed communication protocol between the FPGA and the ADC has a direct impact on the digital coherence of the radar system, potentially leading to increased variance in τ_{dig} and a degraded range resolution as discussed in Section 2.1.2.

2.2.2.1 The JESD204B Protocol

One popular communication protocol between the FPGA and ADC is the JEDEC JESD204B protocol [9]. JESD204B utilizes differentially-fed serial lanes operating at speeds up to 12.5 Gbps to communicate data between devices. The serial lanes are much easier to route on a PCB than a parallel bus, and JESD204B encodes data using an 8b/10b scheme, allowing clock recovery from the data and eliminating the need for a separate data clock line.

To allow for data communication without a clock line, the interface uses a low-speed signal called the SYSREF to set up and align an internal clock in each device called the Local Multiframe Clock (LMFC). The LMFC times the transmission and reception of large groups of samples called multiframes. Each multiframe consists of K frames of samples, and each frame consists of F octets (bytes) worth of sample data per lane. Each sample of data contains N bits per sample and the data is communicated over L different serial lanes. The number of data converters (i.e. ADCs and DACs) taking advantage of the serial link is M, and the number of samples transmitted by each converter in a single frame is S. The bit rate on each lane (f_L) is generally related to the sampling frequency of the data converter. In many cases, the parameters given above are set for a particular data communication application, with only K being defined by the user.

The frequency of the LMFC (f_{LMFC}) is then given in [10] by

$$f_{\text{LMFC}} = \frac{f_s}{K \times S \times M} \quad . \tag{2.15}$$

To achieve the desired LMFC, the SYSREF signal must be configured to the desired LMFC frequency. The LMFC controls the timing of all groups of samples between the ADC and FPGA in a JESD204B link, and as such, a new data capture must begin on the rising edge of the LMFC. It follows that the trigger signal to the FPGA initiating a data capture must deterministically align with the edge of the LMFC. Otherwise, there will be a non-deterministic latency between the trigger signal being issued and capture taking place, leading to digital incoherence discussed in Section 2.1.2.

Chapter 3

Initial System Design

The work done in this thesis is based on an original radar design developed in [11] at the University of Oklahoma. In this work, an RF analog front-end is designed, and components of a digital back-end are selected. This chapter contains an explanation of the original system's theory of operation, including a brief discussion of the analog front-end, the digital components of the RF transceiver, and the digital control system. With respect to the digital back-end section of this chapter, the focus is on system-level requirements driving the selection of hardware. The specific elements of the digital hardware implementation are discussed in further detail in Chapter 4.

3.1 Theory of Operation

The radar is designed to be used in a synthetic aperture radar (SAR) application on board a lightweight aircraft. The radar operates at Ku-band with a carrier frequency of 16.6 GHz. The waveform is an LFM pulse with 300 MHz of bandwidth and a pulse width of 1 μs transmitted with 2 W of power. The bandwidth of the waveform results in a range resolution of 0.5 m at boresight using (2.3), and in a SAR application with a 30° look angle it results in a cross-track resolution of 1 m.

To form an image with this resolution, the radar must capture pulses over a synthetic aperture length of 253.6 m, which is accomplished over the course of 4.5 seconds of flight time by a Piper PA-28-161 Warrior III flying at 57 m/s. In a typical monostatic radar application, the 1 μs pulse width would result in a 150 meter blind range. However, this radar system utilizes a quasi-monostatic configuration with two antennas with high isolation, allowing for simultaneous-transmit-and-receive (STaR) operation, which eliminates the blind range.

The system operates with a PRF of 3 kHz. This frequency was chosen to absorb the entire Doppler bandwidth of a stationary SAR scene observed at an altitude of 1.5 km with a look-angle of $30°$ and an aircraft velocity of 57 m/s.

This section is concerned with the original design of the Ku-band radar system. The original radar block diagram is shown in Figure 3.1, and an image of the full radar hardware is shown in Figure 3.2. The theory and design of both the RF front-end and the digital back-end are detailed in the following sections.

3.2 The RF Front-End

The RF front-end accepts the signal from the digital back-end as an LFM pulse with 150 MHz of bandwidth centered at 1 GHz. The LFM pulse is amplified and upconverted by a x2 frequency multiplier to 300 MHz of bandwidth at 2 GHz. The harmonics of this multiplication are then filtered and the signal is mixed up to the carrier frequency of 16.6 GHz by an LO of 14.6 GHz generated by a dielectric resonating oscillator (DRO). The output RF signal is then amplified to a 2 W transmit power and transmitted by the transmit antenna. It should be noted that the 2 W amplifier is not pulsed: as such, the pulsing of the waveform is accomplished digitally rather than at the output of the RF transmitter.

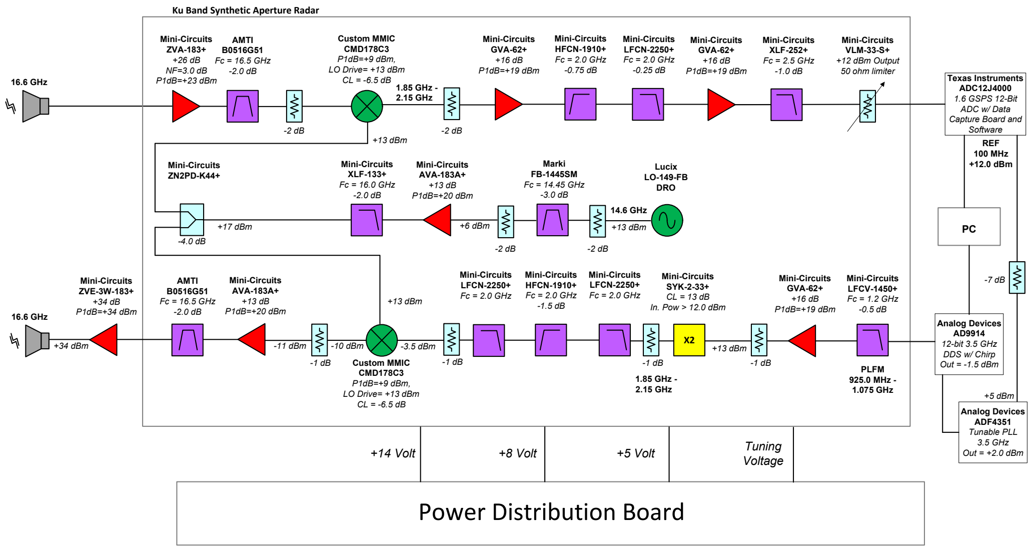

Ku Band Synthetic Aperture Radar
16.6 GHz
Mini-Circuits ZVA-183+ +26 dB NF=3.0 dB P1dB=+23 dBm
AMTI B0516G51 Fc = 16.5 GHz -2.0 dB
-2 dB
Custom MMIC CMD178C3 P1dB=+9 dBm, LO Drive= +13 dBm CL = -6.5 dB
1.85 GHz - 2.15 GHz
-2 dB
Mini-Circuits GVA-62+ +16 dB P1dB=+19 dBm
Mini-Circuits HFCN-1910+ Fc = 2.0 GHz -0.75 dB
Mini-Circuits LFCN-2250+ Fc = 2.0 GHz -0.25 dB
Mini-Circuits GVA-62+ +16 dB P1dB=+19 dBm
Mini-Circuits XLF-252+ Fc = 2.5 GHz -1.0 dB
Mini-Circuits VLM-33-S+ +12 dBm Output 50 ohm limiter
Texas Instruments ADC12J4000 1.6 GSPS 12-Bit ADC w/ Data Capture Board and Software
REF 100 MHz +12.0 dBm
+13 dBm
Mini-Circuits ZN2PD-K44+
-4.0 dB
+17 dBm
Mini-Circuits XLF-133+ Fc = 16.0 GHz -2.0 dB
Mini-Circuits AVA-183A+ +13 dB P1dB=+20 dBm
+6 dBm
-2 dB
Marki FB-1445SM Fc = 14.45 GHz -3.0 dB
-2 dB
14.6 GHz
+13 dBm
Lucix LO-149-FB DRO
PC
-7 dB
+13 dBm
16.6 GHz
+34 dBm
Mini-Circuits ZVE-3W-183+ +34 dB P1dB=+34 dBm
AMTI B0516G51 Fc = 16.5 GHz -2.0 dB
Mini-Circuits AVA-183A+ +13 dB P1dB=+20 dBm
-11 dBm
-1 dB
-10 dBm
Custom MMIC CMD178C3 P1dB=+9 dBm, LO Drive= +13 dBm CL = -6.5 dB
-3.5 dBm
-1 dB
Mini-Circuits LFCN-2250+ Fc = 2.0 GHz
Mini-Circuits HFCN-1910+ Fc = 2.0 GHz -1.5 dB
Mini-Circuits LFCN-2250+ Fc = 2.0 GHz
-1 dB
1.85 GHz - 2.15 GHz
Mini-Circuits SYK-2-33+ CL = 13 dB In. Pow > 12.0 dBm
X2
+13 dBm
-1 dB
Mini-Circuits GVA-62+ +16 dB P1dB=+19 dBm
Mini-Circuits LFCV-1450+ Fc = 1.2 GHz -0.5 dB
PLFM 925.0 MHz - 1.075 GHz
Analog Devices AD9914 12-bit 3.5 GHz DDS w/ Chirp Out = -1.5 dBm
+5 dBm
Analog Devices ADF4351 Tunable PLL 3.5 GHz Out = +2.0 dBm
+14 Volt
+8 Volt
+5 Volt
Tuning Voltage
Power Distribution Board

Figure 3.1: The original radar block diagram

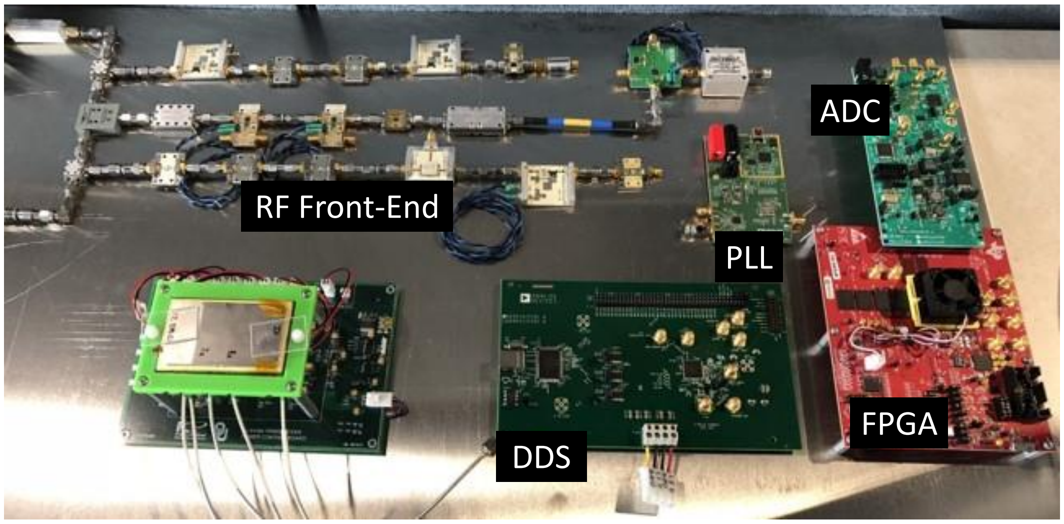


Figure 3.2: The original radar hardware

The receiver accepts the reflected LFM signal with a minimum detectable signal (MDS) at -110 dBm and an input saturation power of -30 dBm. The signal is amplified by a low-noise amplifier (LNA) and filtered, and then mixed back down to an IF of 2 GHz using the same LO as the transmitter. The IF signal is further amplified and filtered, and the final signal is input directly to the ADC at 2 GHz.

3.3 The Digital Back-End

The digital back-end circuitry is selected with the goal of incorporating and testing state-of-the-art digital transceiver hardware. The digital signal generator is the Analog Devices AD9914 DDS with integrated 12-bit DAC, which has a maximum reconstruction frequency of 3.5 GHz [12]. The receiver subsystem is the Texas Instruments ADC12J4000 ADC, which has a maximum sampling frequency of 4 GSPS [1] with 12-bit sampling resolution. The ADC mates with the Texas Instruments TSW14J56 FPGA capture board through a high pin-count (HPC) FPGA mezzanine card (FMC) connector [13]. The capture board contains an Altera Arria V GZ FPGA and 4 GB of onboard RAM. Data is collected from the ADC by the FPGA using the JESD204B interface through the FMC connector, and is then offloaded to a PC for post-processing through a USB cable. For simplicity, each digital device is purchased on an evaluation board to allow the immediate development of a system.

The DDS is operated at the maximum frequency of 3.5 GHz. Operating at a higher frequency pushes the harmonic outputs of the DDS to a higher frequency, making it easier to filter. The DDS generates the 150 MHz LFM pulse over the course of 1 μs, which is then multiplied to 300 MHz of bandwidth in the RF front-end. As stated in Section 3.2, the RF amplifier is not pulsed. As such, the amplitude window of the pulse must be applied by the DDS.

To save FPGA memory during the capture of the waveform, the ADC is operated at a sampling frequency of 1.6 GSPS. This undersamples the 2 GHz waveform from the IF of the receiver, resulting in the captured signal being an aliased signal at 400 MHz. Because the sampling frequency is still greater than twice the bandwidth of the signal per the Nyquist Theorem, the full signal spectrum is preserved.

To derive the digital clocks used in the system, a 100 MHz crystal oscillator on the ADC evaluation board is multiplied by various phase-locked loops (PLLs) to the desired frequencies. The ADC evaluation board utilizes two onboard PLLs to synthesize the sampling clock and the JESD204B clocks: the TRF3765 and the LMK04828 [14, 15, 16]. The precise operation of these devices will be discussed in Section 4.3. The DDS contains an internal PLL that can be used to synthesize a clocking frequency internally; however, it has a maximum clock output of 2.5 GHz, meaning that the 3.5 GHz clock must be synthesized off-chip by an external PLL. The PLL selected for this is the Analog Devices ADF4351 [17]. The 100 MHz clock reference on the ADC evaluation board is tapped off to the ADF4351, which multiplies the signal by x35 to obtain the necessary clocking frequency.

In the initial back-end design, an NI DAQ USB-6251 was used as a controller platform to generate the PRF signal, simultaneously triggering the DDS to synthesize a single pulse and the FPGA to begin saving samples from the pulse on each cycle of the PRF [18]. The configuration of each device is implemented using a USB interface to each evaluation board. The original digital back-end diagram is shown in Figure 3.3.

3.3.1 Evaluation Software

Because the digital components are purchased as evaluation boards, it is convenient to utilize evaluation software to interface with them. Each digital component (with the exception of the FPGA, discussed later) is internally configured by a set

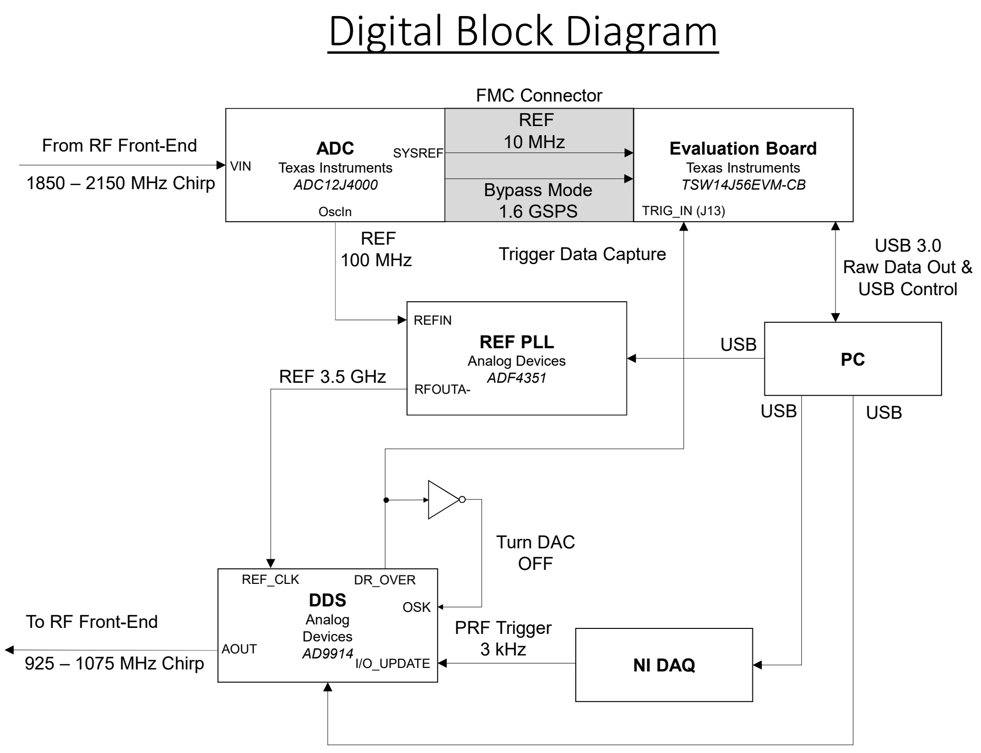


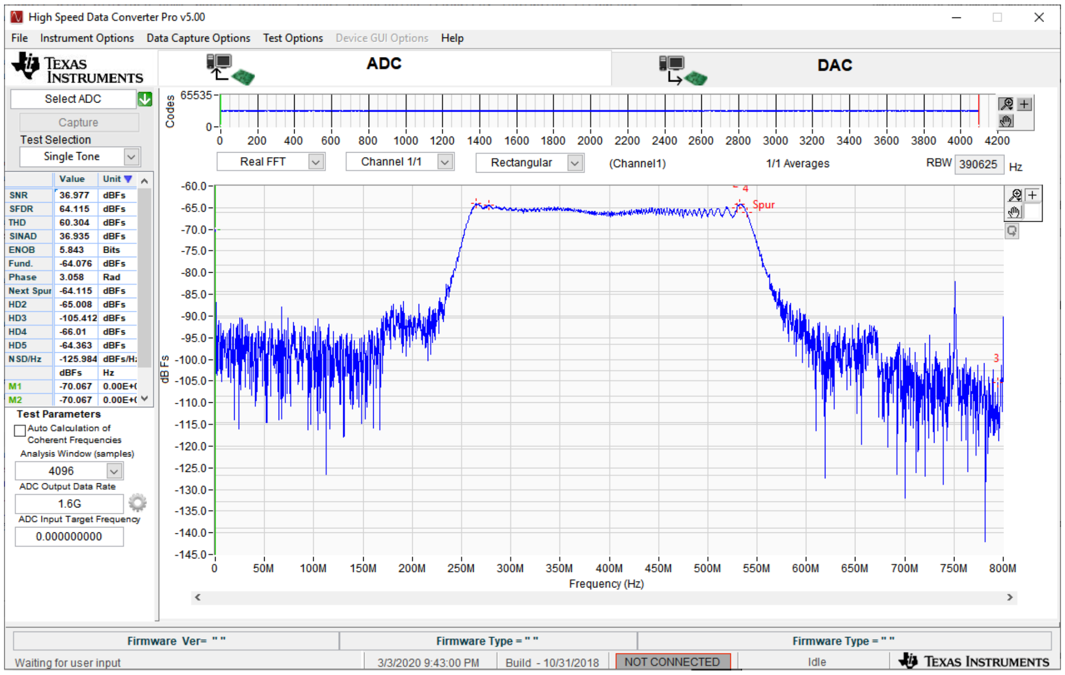
Figure 3.3: The original digital back-end block diagram

of registers, and the registers are written to through an SPI interface. The evaluation boards each have an SPI programmer chip that is controlled by a user's PC through USB. The evaluation software is essentially a user-friendly way of interfacing with the devices, allowing for changes in device operation to be made without a full understanding of the device registry (although register level interface is also available through the evaluation software). Because the hardware can be interfaced directly through SPI, the evaluation software is used with the expectation that an SPI interface will eventually be developed to program all the devices on startup with one single piece of software, rather than using a different piece of software for each device.

To interface with the FPGA, a piece of software developed by TI called *HSDC*

Pro is used. The HSDC Pro GUI is shown in Figure 3.4. HSDC Pro allows for firmware to be ported to the FPGA and it provides an interface for loading captured data from the FPGA RAM into the computer over a USB cable. The software also provides some basic digital signal processing features, including frequency domain analysis and tapering windows. Raw data can also be exported from HSDC Pro in a comma-separated value (CSV) or raw binary file for further processing in MATLAB.

The software is useful in that it enables the basic use of the FPGA without writing custom firmware. The major drawback to HSDC Pro in a pulse-Doppler application is that it does not allow for pulsed data captures. Instead, the software only allows for the manual retrieval of a single data capture at a time. In conjunction with the unmodified FPGA firmware shipped with the capture board, the capture of radar pulses is only possible through the continuous capture of samples. At a sample



Test Selection: Single Tone

	Value	Unit
SNR	36.977	dBFs
SFDR	64.115	dBFs
THD	60.304	dBFs
SINAD	36.935	dBFs
ENOB	5.843	Bits
Fund.	-64.076	dBFs
Phase	3.058	Rad
Next Spur	-64.115	dBFs
HD2	-65.008	dBFs
HD3	-105.412	dBFs
HD4	-66.01	dBFs
HD5	-64.363	dBFs
NSD/Hz	-125.984	dBFs/Hz
	dBFs	Hz
M1	-70.067	0.00E+0
M2	-70.067	0.00E+0

Test Parameters

☐ Auto Calculation of Coherent Frequencies

Analysis Window (samples): 4096

ADC Output Data Rate: 1.6G

ADC Input Target Frequency: 0.000000000

Real FFT | Channel 1/1 | Rectangular | (Channel1) | 1/1 Averages | RBW 390625 Hz

Firmware Ver= " " | Firmware Type = " " | Firmware Type = " "

Waiting for user input | 3/3/2020 9:43:00 PM | Build - 10/31/2018 | NOT CONNECTED | Idle | TEXAS INSTRUMENTS

Figure 3.4: The HSDC Pro GUI

rate of 1.6 GHz and 2 bytes of memory required for each sample, the FPGA memory will fill up after 1.25 seconds, falling short of the 4.5 seconds required to form a full imaging CPI [11]. Solving this problem requires some minor modifications to the FPGA firmware to allow pulsed data captures without pulsed interfacing with HSDC Pro.

3.4 Summary

This chapter presents the initial design of the radar system, including its theory of operation and a high-level overview of the RF front-end and the digital back-end hardware. Before the work done in this thesis was performed, the RF front-end was well-developed, while the hardware for the digital back-end was selected but not connected or tested. Therefore, to begin using the radar as a complete system, the hardware for the digital back-end must be connected, configured, and tested.

Chapter 4

Digital System Configuration

In this chapter, the implementation details of the digital system described in Section 3.3 are elaborated upon. Although the top-level hardware for the digital transceiver was selected in [11], the full operation of the system requires much more involved digital design and the development of additional hardware. In the preliminary stages of implementing the digital radar transceiver, anomalies in the frequency spectra of the LFM pulses were observed, indicating improper waveform synthesis. This problem is also presented in detail in this chapter, along with the preliminary and permanent solution. Some of the modifications to the digital system detailed in this chapter are implemented in the final operation of the radar system - the others are presented in Chapters 5 and 6.

4.1 PLL Configuration

Before the DDS can be used, it must be clocked by an appropriate reference frequency. The DDS clock frequency is to be derived from the 100 MHz oscillator onboard the ADC evaluation board available through the *OSC IN* SMA port. This port is intended to be used as an external clock input in the event that the 100 MHz oscillator is not suitable for a given application; however, the 100 MHz signal is available as an output from this port as well. It is desired to multiply the 100 MHz clock up to 3.5 GHz by using the ADF4351 PLL from Analog Devices.

The device operation is fairly simple, and the evaluation software makes it simple to configure the PLL without a detailed knowledge of the device registry. A simplified diagram of the PLL architecture is shown in Figure 4.1. The PLL contains support for FRAC-N clock manipulation, but this feature is not used and is omitted from this analysis. The PLL takes an input signal at frequency REF_{IN} and divides it down to a frequency f_{PFD} to the input of a phase-frequency detector (PFD). The output of the PFD is low-pass filtered by a loop filter and used to bias a voltage-controlled oscillator (VCO) which generates a much higher frequency f_{VCO}. The output of the VCO is divided by N and used as closed-loop feedback to the PFD, and the PFD eventually settles to an output which biases the VCO such that the divided-by-N version of f_{VCO} matches f_{PFD} in both frequency and phase, at which point the VCO output is "locked." The output of the VCO is then sent through an output divider stage D_{RF} giving the final phase-locked and multiplied signal RF_{OUT}.

The value of f_{PFD} is given by

$$f_{\text{PFD}} = REF_{IN}\frac{D+1}{R(T+1)} \tag{4.1}$$

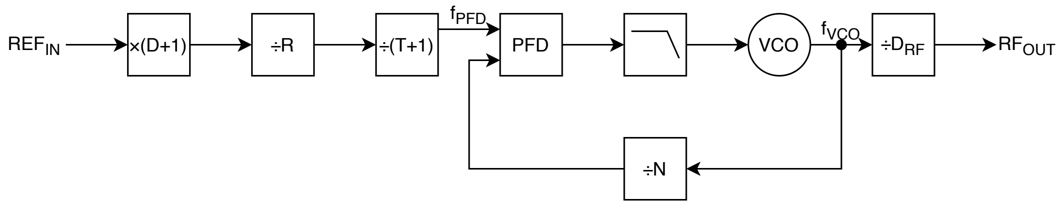


Figure 4.1: A simplified diagram of the ADF4351 PLL architecture

where D and T are bits controlling an optional x2 multiplier and a /2 divider at the input, and R is a prescaling integer divider that can be selected between 1 and 1023 [17]. The PFD requires an input of less than 32 MHz. The output of the PLL RF_{OUT} has a frequency

$$RF_{OUT} = f_{\text{PFD}}\frac{N}{D_{\text{RF}}} = REF_{IN}\frac{N(D+1)}{D_{\text{RF}}R(T+1)} \quad . \tag{4.2}$$

To ensure that f_{PFD} is sufficiently low, R is set to 4 to divide the 100 MHz RF_{IN} to 25 MHz. Setting D and T to 0, D_{RF} to 1, and N to 140 results in an output frequency RF_{OUT} of 3.5 GHz.

The output of the PLL is differential, and the evaluation board breaks the differential leads into two single-end SMA outputs $RF_{OUT}A+$ and $RF_{OUT}A-$. $RF_{OUT}A-$ is terminated in a 50 Ω load, and $RF_{OUT}A+$ is connected to the ***REF IN*** port on the DDS to clock it at 3.5 GHz.

4.2 DDS Configuration

The ADC evaluation board and software have several pre-defined configuration files enabling application-agnostic operation at several different sampling frequencies (including 1.6 GHz). Because the ADC does not require fine-tuning to enable the system to start running, the primary task in setting up the digital back-end is using the DDS to correctly synthesize the LFM pulse. The key constraint that the DDS must fulfill is the synthesis of a 150 MHz LFM pulse at a 1 GHz center frequency with the output of the DDS windowed to 1 μs. This synthesis must be accomplished assuming a 3.5 GHz input as the reference clock generated by the external ADF4351 PLL. Additionally, the DDS must produce this LFM pulse each time it is triggered externally by the 3 kHz PRF. Although the evaluation software can be used to configure the device to operate as desired without an intimate knowledge of the register-level device operation, the USB control of the DDS must be disabled in order to interface with the required control pins to generate the pulses correctly. As such, an SPI protocol must be used (discussed more in Section 4.4) to program the DDS, and the programming must be accomplished at the register level. As such, the operation and use of the digital registry within the DDS is discussed in detail below.

4.2.1 LFM Pulse Generation

To generate the LFM pulse, a feature of the AD9914 called the Digital Ramp Generator (DRG) is employed. The DRG allows the controlled linear sweeping of the frequency, phase, or amplitude of the output waveform, enabling frequency, phase, or amplitude modulation. Using the DRG to sweep over frequency enables the generation of a 150 MHz LFM pulse at 1 GHz by sweeping from 925 MHz up to 1075 MHz.

The output frequency of the AD9914 is controlled by a *frequency tuning word* (FTW) 32-bit register. The FTW register is computed similarly to θ_{inc} in (2.14) for a given output frequency f_{OUT} and a DDS clock frequency f_{DDS} by

$$FTW = \text{round}\left(2^{32}\left(\frac{f_{\text{OUT}}}{f_{\text{DDS}}}\right)\right) \tag{4.3}$$

where the FTW becomes the number of cycles per sample scaled to fill a 32-bit integer. The DRG is configured by giving a lower frequency limit FTW f_{LO}, a higher frequency limit FTW f_{HI}, a frequency step step size Δf, and a ramp rate Δt. Given these configuration values, the DRG will begin the frequency sweep with an output frequency of f_{LO} and will increase in frequency by Δf Hz every Δt s. The frequency sweep will end once f_{HI} is reached by the sweep, and in its default state, the DDS will continue outputting the f_{HI} until the DRG is reset to f_{LO}, at which point the sweep will restart. The registers representing f_{LO} and f_{HI} are given by FTW_{LO} and FTW_{HI}, respectively, and are solved for using (4.3). The 32-bit register value representing Δf is STEP_P and is given by

$$\text{STEP}_P = \text{round}\left(2^{32}\frac{\Delta f}{f_{\text{DDS}}}\right) \tag{4.4}$$

while the 16-bit register value representing Δt is P and is given by

$$P = \Delta t\frac{f_{\text{DDS}}}{24} \quad . \tag{4.5}$$

The factor of 24 in (4.5) is because the internal timer clock governing the DDS operations and the steps of the sweep runs at the frequency of the DDS sampling clock divided by 24. That is, the timer clock runs at a frequency f_{timer} given by

$$f_{\text{timer}} = \frac{f_{\text{DDS}}}{24} \quad . \tag{4.6}$$

The effects of this clock on system coherence are discussed later.

The number of steps required to complete a sweep can be solved for both as a function of the sweep bandwidth and the pulse length. That is,

$$\text{\# Steps} = \frac{f_{\text{HI}} - f_{\text{LO}}}{\Delta f} = \frac{T_p}{\Delta t} \tag{4.7}$$

where T_p is the pulse length discussed in Section 2.1. To give the most continuous sweep, Δt is assumed to be the smallest possible value. This is achieved when

$P = 1$, leading to $\Delta t = \frac{24}{f_{\text{DDS}}}$. Rearranging the right-hand side of (4.7) to solve for Δf yields

$$\Delta f = \frac{\Delta t \left(f_{\text{HI}} - f_{\text{LO}}\right)}{T_p} = \frac{24 \left(f_{\text{HI}} - f_{\text{LO}}\right)}{f_{\text{DDS}} T_p} \quad . \tag{4.8}$$

Given $f_{\text{LO}} = 925$ MHz, $f_{\text{HI}} = 1075$ MHz, $f_{\text{DDS}} = 3.5$ GHz, and $T_p = 1$ μs, Δf is determined to be 1.0286 MHz. Using $P = 1$ and rearranging (4.5) for Δt yields $\Delta t = 6.8571$ ns. Using (4.3)-(4.5), the register values required for the desired DDS configuration are solved for. These configuration values are given in Table 4.1.

Once the DRG is configured to produce the LFM waveform, the signal must then be windowed at the DDS output. Because the RF front-end does not utilize a pulsed amplifier, this windowing must be performed in the DDS. To perform the output windowing, three features of the DDS are utilized: the output shift keying (OSK) function, the DRG over (*DROVER*) digital output, and the "autoclear digital ramp accumulator" function.

The OSK feature of the DDS, when enabled, allows the user of the DDS to switch the output amplitude of the DDS back and forth between two different states. The OSK is controlled by an external *OSK* pin. When the *OSK* pin is held logic low, the output amplitude is set to 0. When it is held logic high, the output amplitude is set to a pre-programmed amplitude value controlled by a 12-bit amplitude profile. The amplitude register is simply set to 0xFFF to effectively set the output amplitude to 1 when the *OSK* pin is held high.

Parameter	Register Name	Parameter Value	Register Value (Hex)
f_{LO}	FTW_{LO}	925 MHz	0x43A83A83
f_{HI}	FTW_{HI}	1075 MHz	0x4EA0EA0E
Δf	STEP$_P$	1.0286 MHz	0x00134272
Δt	P	6.8571 ns	0x0001

Table 4.1: The register values used to configure the DDS DRG

In order to window the output of the DDS correctly using the OSK feature, a 1 μs square pulse must be applied to the *OSK* pin at precisely the time the DRG is sweeping up in frequency. To accomplish this, the DDS is configured to apply a "self-window" using the *DROVER* pin. The *DROVER* is a digital output pin that, when enabled, is held high by the DDS once the DRG has completed a sweep. That is, for this application, the *DROVER* pin will be held at logic low during the generation of the LFM pulse, and will then alternate to logic high once the frequency sweep has concluded. Theoretically, this produces a signal that is logic low for exactly the 1 μs pulse length during the generation of the pulse. To obtain the correct polarity pulse to be used as an input to the *OSK* pin, the output of the *DROVER* pin is passed through a 7404 inverter DIP chip on a breadboard. This inverted pulse is a 1 μs logic high pulse centered (ideally) around the LFM ramp generated by the DRG.

Because the DRG holds the output frequency at f_{HI} at the end of the frequency sweep and *DROVER* is held high indefinitely afterward, the output of the DDS will remain off until the DRG is somehow reset to f_{LO}. Therefore, to actually window the output of the DDS, a mechanism must be set in place to trigger the DRG to begin a new sweep when a new pulse is desired to be generated. This is accomplished using the "autoclear digital ramp accumulator" function, or autoclear. When this feature is enabled, the accumulator used to keep track of the output frequency of the DRG is reset when a digital input pin called the *IOUPDATE* observes a rising edge. That is, when a rising edge is applied to *IOUPDATE*, the frequency sweep is reset. Therefore, if a pulse trigger is input to *IOUPDATE*, the DDS will output a single microsecond pulse. A diagram demonstrating the signal states relative to one another during a pulse is shown in Figure 4.2.

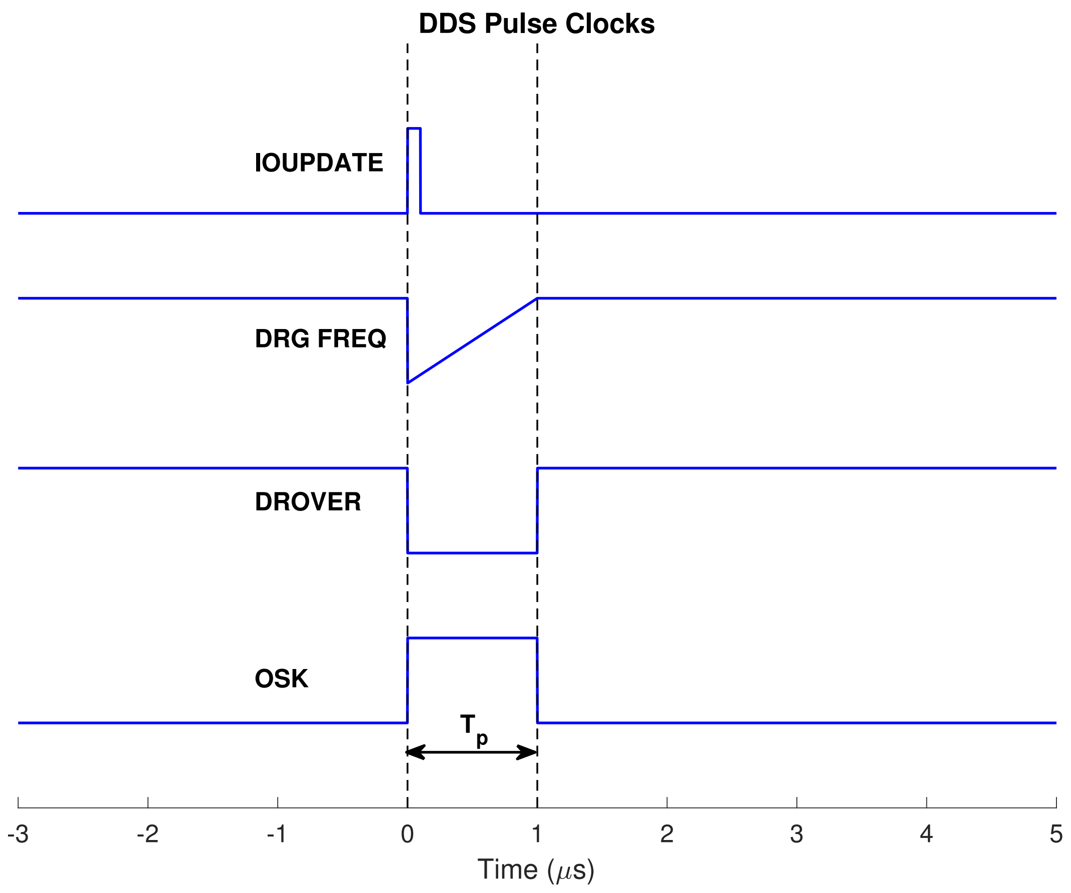


Figure 4.2: A diagram of the DDS control signals during the generation of a pulse

4.2.2 DDS Output Waveform

Using the waveform generation method described in Section 4.2.1, the DDS outputs a radar pulse to the RF front-end. The pulse was measured directly at the output of the DDS to determine the quality of the signal. The time-domain pulse and its associated spectral content are shown in Figure 4.3. It can clearly be seen that there is an amplitude bounce problem at the beginning of the pulse. The pulse appears to turn on briefly before turning back off and then on again. In addition to this, the spectral content of the LFM pulse appears distorted on the higher end of the spectrum, and the roll-off from the main bandwidth of the signal is distorted and not smooth. For reference, the spectrum of an ideal chirp with 150 MHz of bandwidth centered at 1 GHz is shown in Figure 4.4.

To determine the cause of these time-domain and spectral anomalies, it is helpful to observe the instantaneous signal frequency as a function of time during the pulse. The instantaneous frequency is computed for the function using the algorithm described in [19]. The instantaneous frequency of the pulse in Figure 4.3 as a function of time is shown in Figure 4.5.

Viewing the instantaneous frequency around the beginning of the pulse, it appears that the frequency is actually held at f_{HI} for 40 ns until the ramp is restarted,

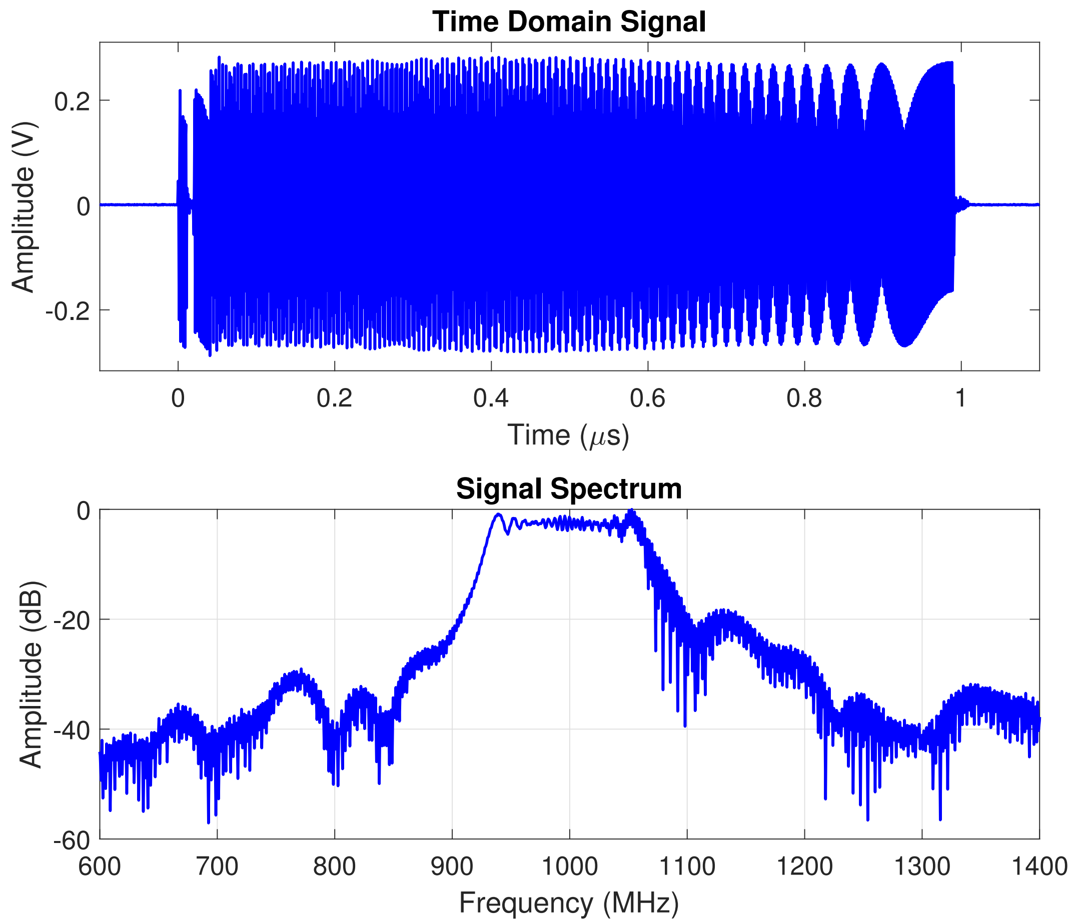


Figure 4.3: The DDS generated pulse (top) and the pulse's spectral content (bottom)

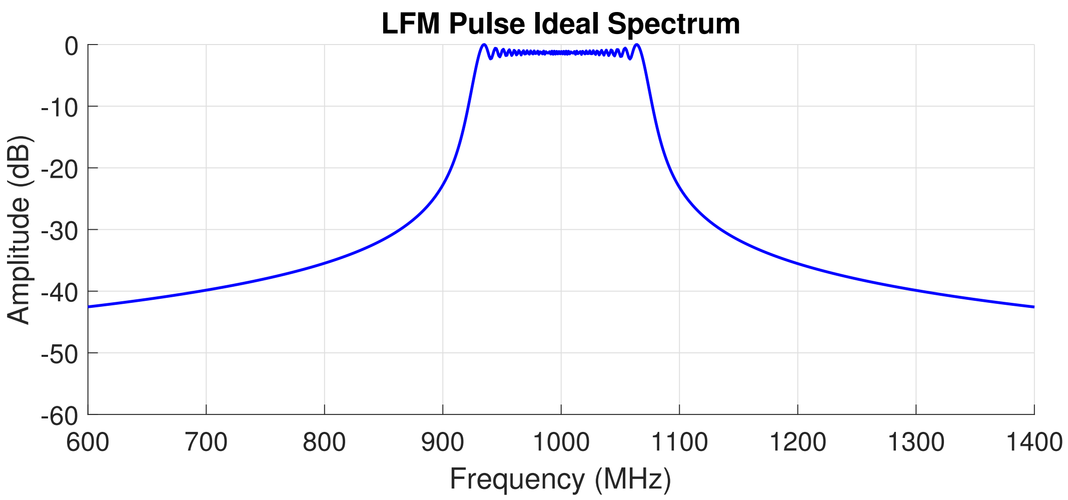


Figure 4.4: The ideal spectrum of a 1 GHz LFM pulse with 150 MHz of bandwidth

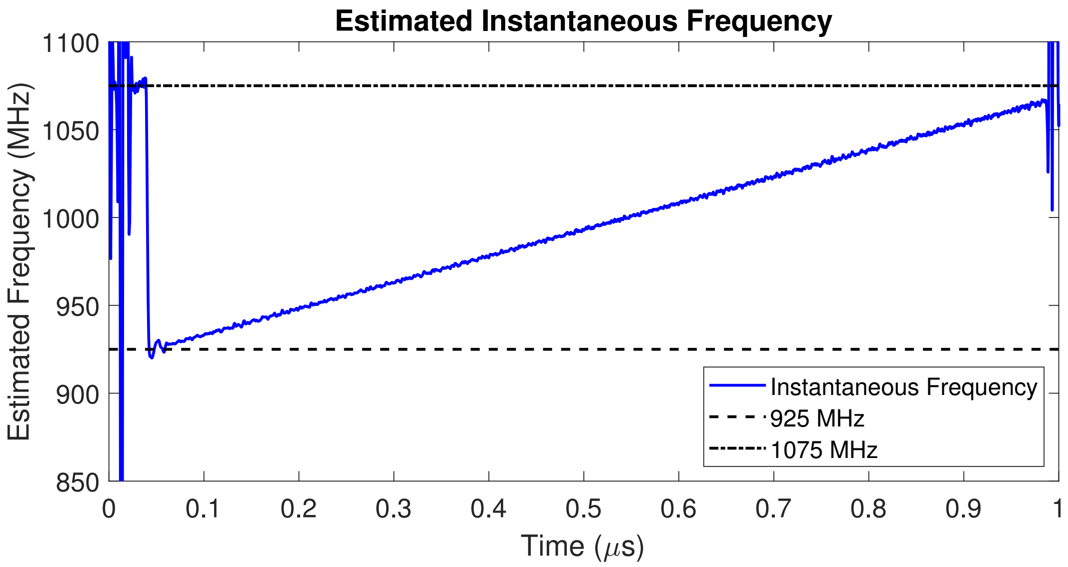


Figure 4.5: The time-varying instantaneous frequency of the LFM pulse

indicating that the OSK amplitude and DRG frequency updates to the output waveform are not implemented simultaneously. The reason for this is that the internal structure of the DDS uses a pipeline architecture. For reasons abstracted from the end user, updates to the frequency, phase, and amplitude of the DDS output are not implemented simultaneously in the default configuration of the device due to varying latency through the DDS pipeline. Fortunately, the DDS has a "matched latency" feature which, when enabled, forces the updates to frequency, phase, and amplitude to be applied to the output waveform simultaneously. This feature is implemented and the signal is regenerated. The resulting signal's time-domain representation, frequency spectrum, and time-varying instantaneous frequency are shown in Figure 4.6.

While this appears to correct for most of the timing and spectral anomalies of the pulse, the LFM waveform is still not windowed perfectly. It can be seen in Figure 4.6 that the spectrum is not symmetric around 1 GHz, but that it is weighted more heavily toward the higher frequencies. From the instantaneous frequency plot, this is evidently because the LFM pulse starts at a slightly higher frequency than 925 MHz and is then held for 70 ns at the high frequency before shutting off. This indicates again that the amplitude window applied to the *OSK* pin is not timed correctly. The loss of the full signal spectrum leads to a degradation of the range resolution of the radar due to a lower bandwidth in (2.3).

Because the matched latency feature has already been implemented, the remaining source of error can only be attributed to the *DROVER*. From [20], it is seen that the DDS series including the AD9914 does not account for the pipeline delay in the *DROVER* signal. As such, it is not suitable for use in the precise timing of the output amplitude of a 1 μs pulse. It follows that the DDS is not capable of implementing the rectangular amplitude window for the pulse by itself. The mechanism used to correctly implement the pulse window is discussed in Section 4.4.

4.3 ADC and FPGA Configuration

For the most part, the ADC is configured entirely by using the evaluation software associated with the evaluation board. However, there are several facets of the

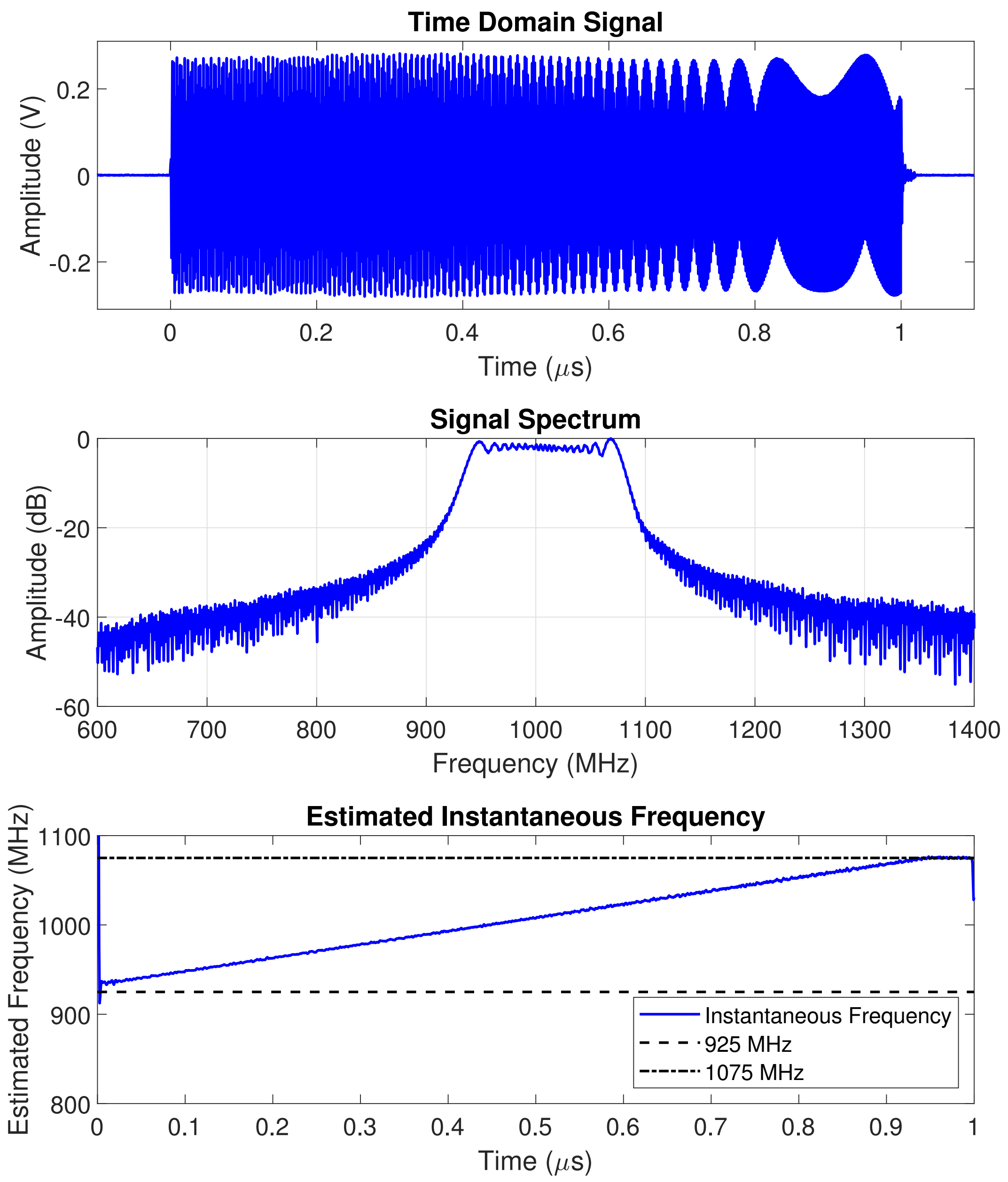


Figure 4.6: The time-domain pulse (top), spectral content (middle), and instantaneous frequency (bottom) of the LFM pulse after "matched latency" implementation

internal operation of the ADC that are useful to understand, particularly with respect to its association with the FPGA. Additionally, the FPGA itself requires some configuration through the HSDC Pro software to enable it to function as desired.

These configuration steps are discussed below.

4.3.1 Basic ADC Configuration

The evaluation software for the ADC12J4000EVM does not allow for quite as precise control of the ADC as the DDS software. The ADC cannot be configured by the software to any arbitrary sampling frequency, but is instead restricted to a short list of preset operational modes. This is in part due to the fact that the ADC evaluation board synthesizes all of its relevant clock frequencies using a 100 MHz oscillator reference onboard rather than relying on an external clock, like the DDS. As such, each different sampling frequency requires configuration details not only for the ADC but the PLLs used to synthesize the relevant clocks as well, making it difficult to implement a continuously reconfigurable sampling frequency. The configuration details for each device (the ADC12J4000, the TRF3765, and the LMK04828) are stored in configuration files which essentially list the hexidecimal values to be loaded into each register on the device and the order in which they are to be loaded. When programming the evaluation board, the evaluation software reads the register values listed in these files and sets each register to its associated value found in the configuration file. Therefore, if the configuration of the devices must be changed beyond the capability of the evaluation software, the registry of the devices can still be modified as needed by altering the configuration files.

Fortunately, a 1.6 GHz sampling frequency is a pre-defined setting in the evaluation software, so to begin operating the ADC at this frequency does not require any modifications to the configuration files. The default 1.6 GHz settings use the TRF3765 to multiply the 100 MHz oscillator by x16 to obtain the clocking frequency to the ADC. The ADC is operated by default in "bypass mode," which refers to bypassing the digital down-conversion (DDC) block in the ADC that would effectively downsample the waveform. The result is that data is recorded in real format rather than as complex baseband data, requiring down-conversion in the post-processing stage in MATLAB.

4.3.2 ADC-to-FPGA JESD204B Configuration

The ADC and FPGA communicate with one another via a JESD204B interface. The interface is implemented in the firmware shipped with the FPGA and is a feature that is built into the ADC. For any particular configuration of the DDC in the ADC, the parameters of the JESD204B interface are mostly fixed with a minor amount of flexibility in selecting the K value, which has a default value of 4. The JESD204B parameters for the ADC12J4000 in bypass mode are given in Table 4.2.

Given a sampling frequency of $f_s = 1.6$ GHz and the parameters listed in Table 4.2, (2.15) gives a local multi-frame clock frequency f_{LMFC} of 10 MHz. When in bypass mode, the ADC also uses a double data rate (DDR) setting to double the clock frequency of the sampling clock to derive the serial lane clock, giving a JESD204B lane frequency of $f_l = 3.2$ GHz.

To derive the 10 MHz LMFC, the SYSREF signal must be derived and used as an input both to the ADC and the FPGA. This signal is synthesized by the LMK04828. The 1.6 GHz clock generated by the TRF3765 is passed through a divide-by-2 buffer to create an 800 MHz signal, which is input to the LMK04828. This 800 MHz signal is internally divided by 80 by the LMK04828 to create the 10 MHz SYSREF signal. The SYSREF clock is then distributed by two discrete outputs of the LMK04828 to the ADC and the FPGA to setup the LMFC on each device.

JESD204B Parameter	Value
N	12
L	8
F	8
M	8
S	5
K	4-32

Table 4.2: Bypass Mode JESD204B Parameters for the ADC12J4000 [1]

4.3.3 FPGA Configuration

The FPGA is configured using the HSDC Pro software discussed in Section 3.3.1. Once connected to the FPGA through USB, HSDC Pro is configured to process the ADC data based on its model and the desired decimation mode. An HSDC Pro configuration is available for the ADC in bypass mode. The configuration information is contained in a .ini file which defines several different parameters of the software and FPGA operation. The primary parameters of note are the FPGA *JESD IP Core* parameters and the *Trigger Input Polarity Selection* parameter.

The JESD IP Core parameters simply define the operation of the JESD204B interface between the FPGA and the ADC, and they are set to match the ADC parameters given in Table 4.2. The Trigger Input Polarity Selection defines whether the digital trigger to the FPGA through the *TRIG IN* pin is active high (1) or low (0). That is, it defines whether the FPGA should begin a data capture on the rising edge or falling edge of the trigger signal. To ensure that the data capture takes place simultaneously with DDS waveform generation, Trigger Input Polarity Selection is set to 1. The trigger generated by the digital control system (see Section 4.4) that is input to *IOUPDATE* on the DDS to trigger a waveform generation is also input to *TRIG IN* on the FPGA, simultaneously initiating waveform synthesis and waveform capture.

To enable the FPGA triggering through hardware, the “Trigger Mode Enable” option must be selected in HSDC Pro. This enables the FPGA to begin the data capture when a signal is applied to *TRIG IN* rather than when an arbitrarily timed trigger (dubbed a “Software Trigger”) is created by HSDC Pro. HSDC Pro also allows the number of captured data samples to be modified. In order to observe an entire pulse, only 4096 samples are required.

4.4 Digital Control System Configuration

Because the DDS and FPGA are not capable of generating a PRF internally or operating synchronously with one another by nature, some form of digital control system is required to synchronize the system and generate control signals, namely the PRF trigger. The original vehicle for the control system was the NI DAQ USB-6251, which was selected based on the DAQ’s ability to coherently capture and generate multiple signals, and the ease of integration with an NI LabVIEW custom control GUI.

There are a few drawbacks to using the NI DAQ. The primary consideration is the poor SWaP performace of the device. The DAQ, shown in Figure 4.7, is quite bulky and heavy, especially compared to alternatives. Additionally, as discussed in Section 4.2, the DDS must be configured using an SPI interface implemented by the digital control system. The NI DAQ does not contain built-in support for SPI, instead requiring a custom SPI driver to be programmed from scratch in LabVIEW. While this is possible, the resulting interface is very slow and unreliable.

An alternative solution to the NI DAQ is to use a microcontroller with a breakout board. Most microcontrollers include an SPI subsystem block in hardware, requiring minimal configuration before use. Additionally, the core microcontroller

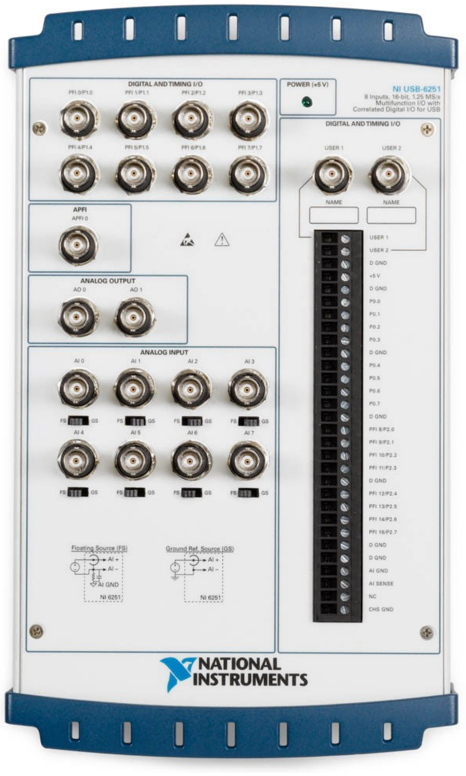


Figure 4.7: The NI DAQ USB-6251 BNC

functions are performed on a single IC, enabling small and light controller boards. The only drawback to using such a controller is the absence of native integration capabilities with LabVIEW. However, most microcontrollers also contain support for RS-232 communication through a USB port, allowing communication with a PC in LabVIEW through the NI VISA feature.

The microcontroller that is initially selected to replace the NI DAQ is the mbed NXP LPC1768 [21]. An image of this microcontroller is shown in Figure 4.8. This controller and its associated API abstract away most of the low-level programming to high-level functions, making it very quick and easy to develop rapid prototypes of the system. The microcontroller contains an onboard 12 MHz crystal oscillator and operates at a clock frequency of 96 MHz. The microntroller must program the DDS using the SPI interface on startup, and it must also generate a 3 kHz PRF signal to trigger the DDS and FPGA.

4.4.1 PRF and Offset Amplitude Window

As discussed in Section 4.2.2, the amplitude windowing of the LFM pulse generated by the DDS cannot be accomplished using any onboard mechanism on the DDS. As such, the trigger to the *IOUPDATE* and *OSK* pins on the DDS must be asserted for exactly 1 μs - this will window the output of the DDS to the desired pulse length. The amplitude window is implemented with the PRF by using the pulse-width modulation (PWM) subsystem of the microcontroller.

PWM is designed to generate square-wave pulses with any arbitrary period and pulse width. PWM operates in most microcontrollers using a hardware timer that resets to 0 once the value in a match register (*MR*) is reached. The PWM period is equal to the value in *MR* plus 1 and multiplied by the clock period of the PWM subsystem, which is often equivalent to the clock period of the microcontroller. The output of each PWM channel is set to high when the timer reaches another channel-specific match register (*MR1*), and is set low when the timer reaches a third match register (*MR2*). In this way, the pulse length, period, and phase of the PWM channels can be controlled with precision. An example of the PWM operation is

Figure 4.8: The NXP LPC1768 microcontroller

shown in Figure 4.9. To generate a 1 μs pulse at a frequency of 3 kHz (or a period of 333.3 μs), the PWM is configured to output a pulse that is off for 31904 clock cycles (332.3 μs at a 96 MHz clock frequency) and on for 96 clock cycles (1 μs), giving a total period consisting of 32000 clock cycles. To accomplish this, *MR* is set to 31999, *MR1* is set to 31903, and *MR2* is set to 31999, causing the PWM channel to turn on for 1 μs at the end of the period.

Although this correctly generates the 1 μs window to be passed to the *IOUPDATE* and *OSK* pins, the instantaneous frequency plot and chirp spectral content still appear similar to those in Figures 4.3 and 4.5, indicating that the amplitude window is still not properly aligned with the DRG reset. To compensate, the double-edged PWM feature is employed. This allows two PWM signals with the same period to be implemented with the rising and falling edges timed differently from one another. When *IOUPDATE* and *OSK* are asserted simultaneously, the DRG does not reset the frequency sweep before the amplitude is turned on. Therefore, the correction must enable *IOUPDATE* to be asserted before *OSK*. Based on the latent amount of time in the pulse that the frequency ramp is not correct, the timing offset should be set to as close to 40 ns as the microcontroller can obtain.

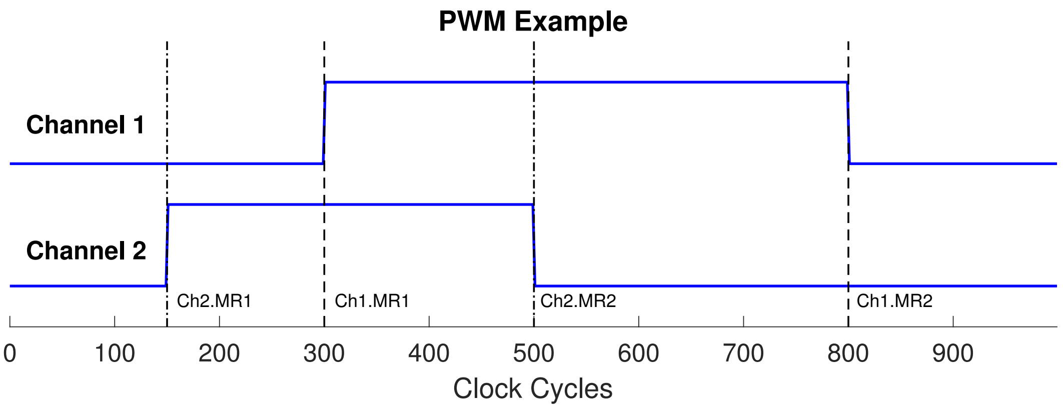


Figure 4.9: A 2-channel PWM example with *MR*$= 999$, Channel 1 *MR1* $= 299$ and *MR2* $= 799$, and Channel 2 *MR1* $= 149$ and *MR2* $= 499$

Two PWM signals with the same period of 32000 clock cycles (*MR* $= 31999$) are configured. The first signal remains off for 31904 clock cycles and turns on for 96 clock cycles as it did originally. That is, for the first channel, *MR1* $= 31903$ and *MR2* $= 31999$. This signal is input to the *OSK* pin to window the pulse output to 1 μs. The second signal remains off for 31900 clock cycles and turns on for 100 clock cycles, meaning that it goes high 4 clock cycles (41.67 ns) before the first signal. That is, for the second channel, *MR1* $= 31899$ and *MR2* $= 31999$. This signal is input to the *IOUPDATE* pin to trigger the DRG to begin a new frequency ramp. The resulting chirp instantaneous frequency and spectral content, shown in Figure 4.10, are linear over the correct frequency range for exactly 1 μs and closely match the ideal spectrum in Figure 4.4.

4.5 Full System Loopback Test

Once the system is verfied to consistently generate a chirp with the correct characteristics and be able to capture the signal for post processing, a full system loopback is performed. This test is performed by connecting the digital back-end to the RF front-end as it would be in normal operation as in Figure 3.1 with a short coaxial cable and an 80 dB attenuator connecting the output of the transmitter with the input of the receiver.

When the system is operating properly, the signal observed at the ADC should be an LFM pulse with 300 MHz of bandwidth centered at 400 MHz. The instantaneous frequency plot should show a linearly increasing frequency starting at 250 MHz and ending at 550 MHz. The signal, shown in Figure 4.11, meets these criteria, indicating that the signal produced and captured by the digital back end is compatible with the RF front-end and meets the spectral requirements of the

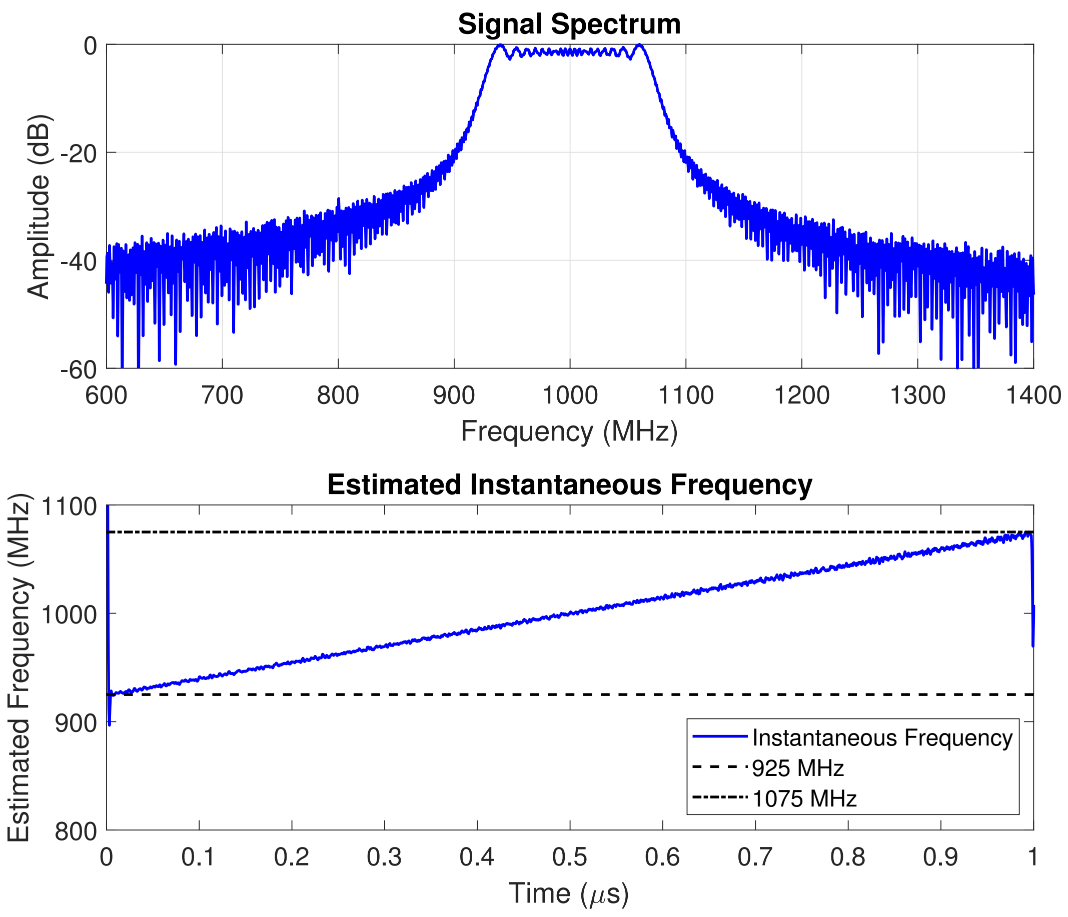


Figure 4.10: The spectral content (top), and instantaneous frequency (bottom) of the LFM pulse when *IOUPDATE* is triggered 4 clock cycles before *OSK*

radar waveform. The waveform is not perfect, as it appears to have some defects in the time and frequency domains, namely a rolloff in ampliude towards the higher frequencies of the pulse. This is almost certainly due to non-ideal frequency response shape in the passband of the filters and amplifiers used in the RF front-end. These defects could potentially be solved using some sort of digital predistortion or automatic gain control techniques, though these features are not pursued here.

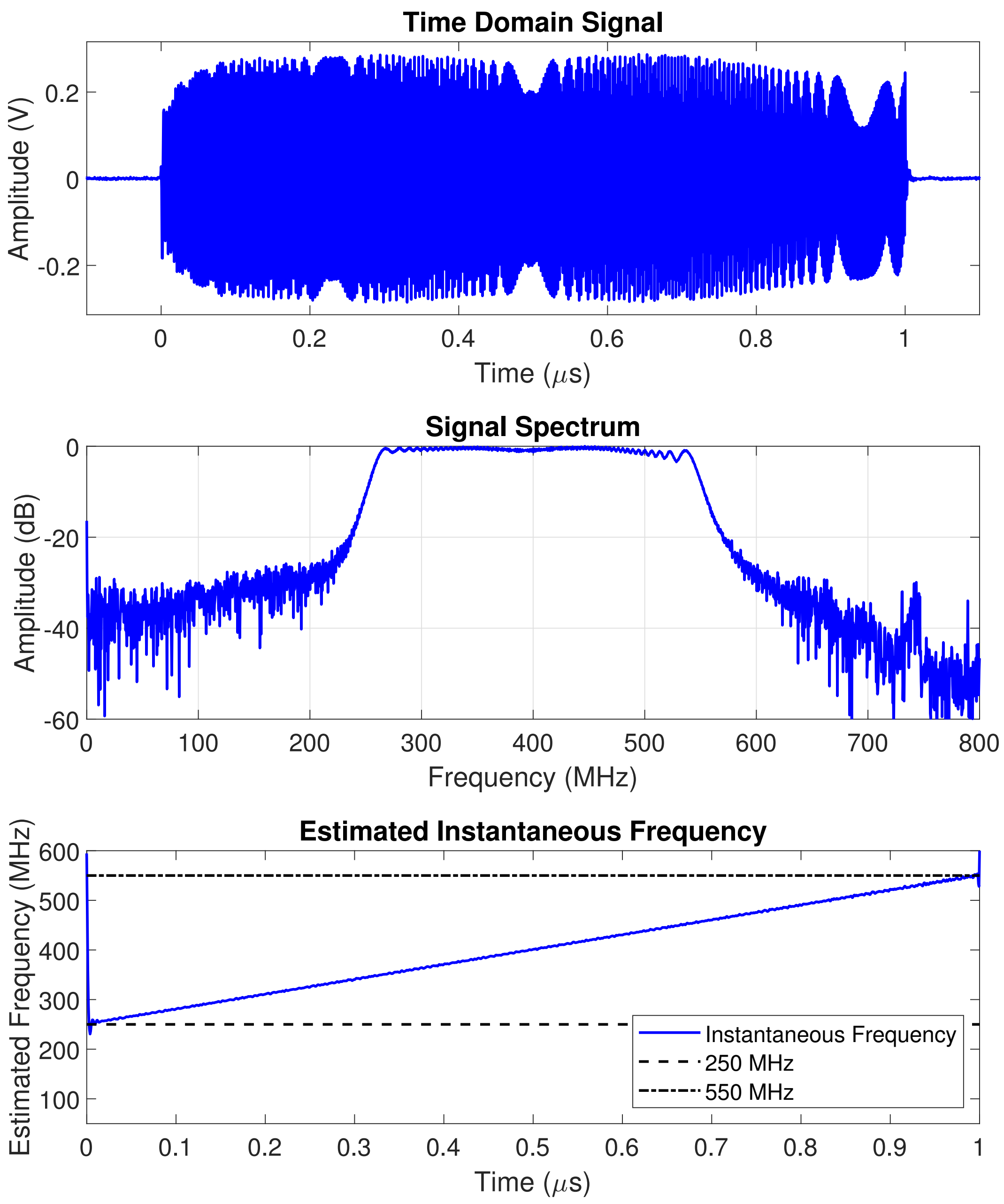


Figure 4.11: The time-domain pulse (top), spectral content (middle), and instantaneous frequency (bottom) of the LFM pulse through the RF front-end

4.6 Summary

This chapter presents the configuration and layout of the digital back-end of the radar. The full block diagram with detailed pinouts for this version of the back-end is shown in Figure 4.12. This initial configuration contains enough implementation details to allow the desired pulsed LFM signal to be generated with the proper spectral properties and to be sampled and saved properly by the ADC and FPGA.

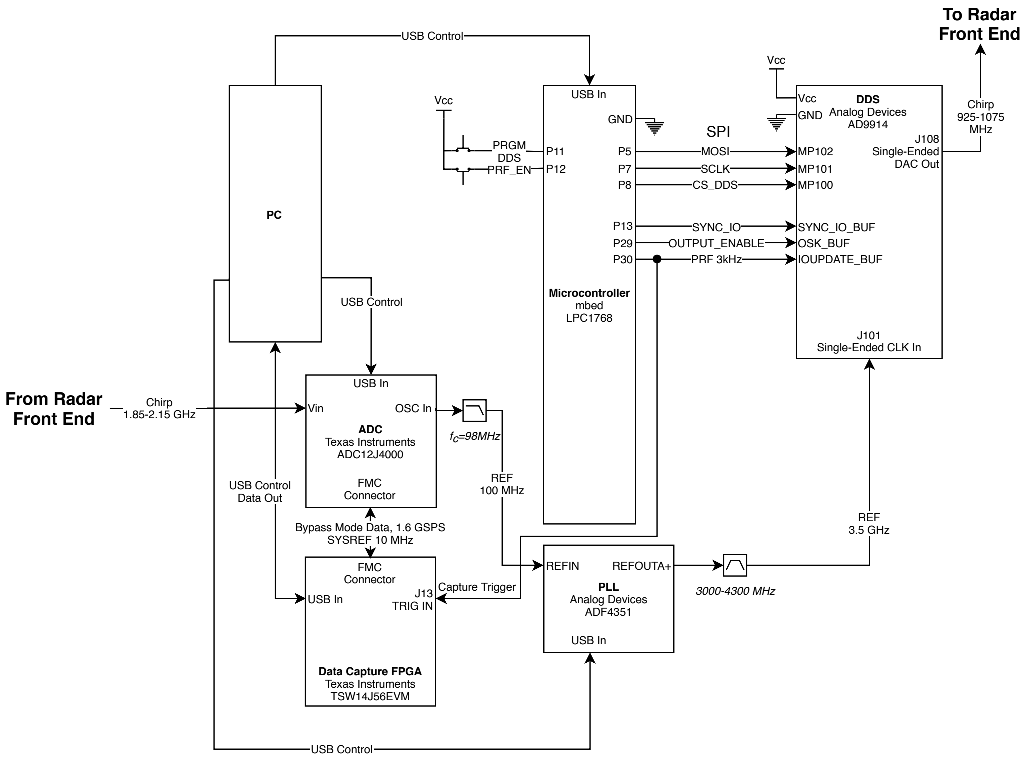


Figure 4.12: A detailed block diagram of the digital back-end

Chapter 5

Pulse-to-Pulse Incoherence

In the previous chapter, the implementation details allowing the digital back-end to generate and capture single pulses with correct waveform characteristics and spectral quality was discussed. The proper signal characteristics enable the predictable signal processing steps of single pulses as discussed in Section 2.1.1. However, single pulses are generally not useful by themselves - a coherently timed pulse train provides gain from coherent integration and allows Doppler information to be extracted and a synthetic aperture to be formed for SAR imaging.

In this chapter, the timing characteristics of the system with respect to coherent pulsing are discussed. Several issues concerning high-speed clock coherence are discussed, leading to decreased range precision as explained in Section 2.1.2. The engineering solutions to those issues are delivered, resulting in modifications to the RF front-end and digital back-end. The changes to the system developed in this chapter result in a radar system that is coherent in timing from pulse to pulse with less than 60 ps of timing jitter on each pulse.

5.1 Timing Jitter from Pulse to Pulse

Given the success in generating properly shaped radar pulses individually, the next step in operating the radar is to determine whether transmitting a coherent pulse train is possible. At the time this part of the project was completed, coherent pulsing capabilities for capturing pulse trains on the FPGA described in Chapter 7 were not available. However, pulses could still be triggered individually by the PRF and captured one at a time to indicate the timing stability of the pulses.

Observering the timing difference between pulses using the radar configured in Chapter 4 shows that there is a large range of variation in pulse latency each time a pulse is captured. Three pulses captured in succession are plotted in time in Figure 5.1. These pulses are transmitted through a loopback cable, implying that their propagation delay from the DDS to the ADC will be constant. Therefore, the pulses should overlap perfectly to indicate perfect coherence in timing and phase. Instead, they are spaced out with a variation of 60 ns. Using (2.13) determines that this will introduce 9 m of ambiguity to any range measurement. At its worst, timing variations up to 100 ns were observed, resulting in a 15 m range ambiguity introduced to measurements. Given that the range resolution of the radar is 0.5 m, this value should preferably be on the order of 5 cm or less. In the following sections, the solutions to this problem are derived.

5.2 Clock Incoherence Between FPGA and PRF Trigger

Timing inconsistency in a digital system is often caused by metastability due to varying misalignment in clock edges governing the system. Generally, in order for clocks in a digital system to be aligned properly and consistently, their frequen-

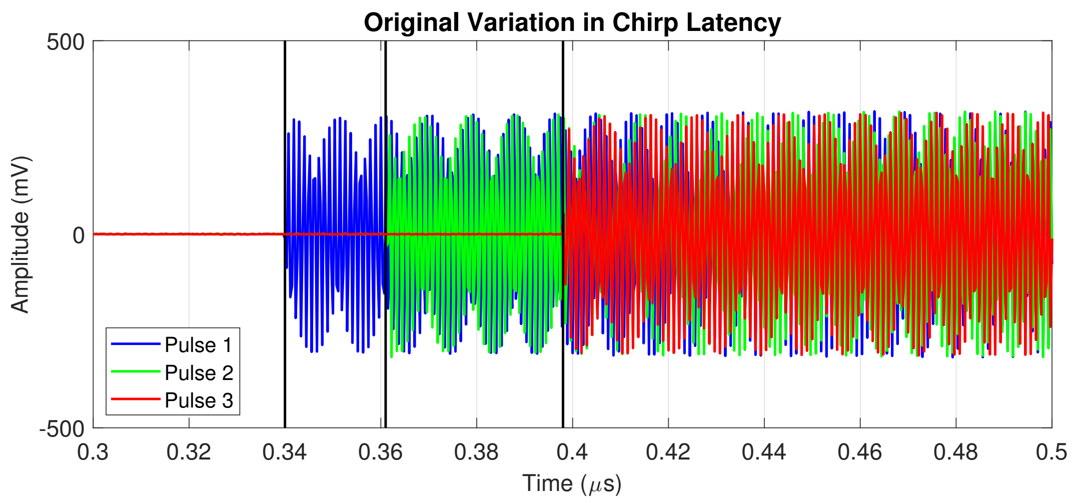


Figure 5.1: Three pulses plotted in time showing 60 ns of variation

cies must be integer multiples of one another and they must be phase-locked to one another through the use of a phase locking mechanism like a PLL. Digital systems without clock distributions like this are referred to as asynchronous. While asynchronous circuits can be useful in low-power and low-speed serial communications, they cannot be used to construct a pulse-Doppler radar system with strict timing requirements.

Because the variance in pulse timing occurs up to 100 ns per pulse, this implies that the variance is due to misalignment with a clock operating with a clock period of 100 ns, or a frequency of 10 MHz. The only clock in the system (apart from the 3 kHz PRF) that operates at such a low speed is the LMFC of the JESD204B interface, which operates at exactly 10 MHz. As explained in Section 2.2.2, the trigger signal issued to the FPGA to begin a data capture must be synchronous with the LMFC; otherwise, the capture will not occur exactly when the trigger is issued. If the trigger edge varies in time relative to the edge of the LMFC, this will lead to a variation in timing of each pulse.

Because the microcontroller generating the 3 kHz PRF triggering the data captures is clocked using its own oscillator, it must be assumed that it cannot generate a synchronous PRF signal with the FPGA LMFC. To determine if the timing variation can be removed or reduced by issuing a synchronous trigger with the LMFC, the digital back-end is disconnected from the PRF signal from the microcontroller. The trigger is then generated using the FPGA "software trigger" feature, which generates a hardware trigger on the FPGA when instructed by HSDC Pro to do so. Because the trigger is generated for the purpose of triggering the FPGA in addition to other devices, the trigger signal is assumed to be triggered synchronously with the LMFC. The trigger is used to initiate the capture on the FPGA but can also be broken off the board through an SMA port (*TRIG OUT*) to be used elsewhere. Since the trigger generated by the FPGA cannot be configured to a desired pulse width and is not 1 μs in length, it cannot be used to window the DDS output like the microcontroller. As such, the DDS is temporarily reconfigured to "self-window" using the *DROVER* feature. Although this will hurt the pulse waveform characteristics, the pulses will still be timed consistently and give an indication of the pulse-to-pulse timing characteristics.

The test setup is summarized in Figure 5.2. As expected, the LFM pulse de-

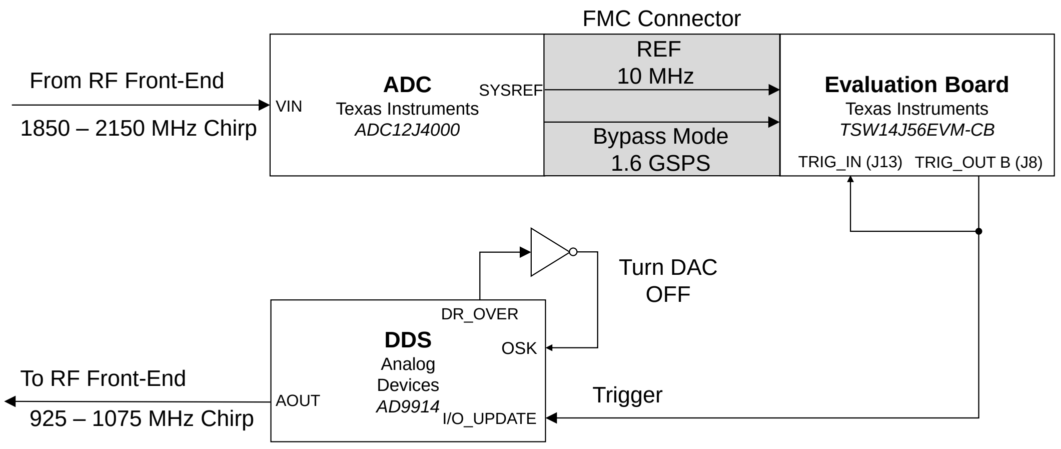


Figure 5.2: A simplified digital block diagram using the FPGA software trigger

grades in quality to that seen in Figure 4.6. However, the variation in pulse latency is reduced in this scheme from 100 ns to close to 7 ns, bringing the range ambiguity down from 15 m to around 1 m. Note that the precise change in latency is difficult to determine because the sample period at 1.6 GHz is 625 ps. Thus, the exact latency variation value is somewhere within the range of 7±0.625 ns. This is a marked improvement, but it is still necessary to remove the remaining variation to make the radar operational. Additionally, this solution to the latency variation cannot be permanent, as it does not allow for the correct pulse windowing like the microcontroller does and cannot be extended to a 3 kHz PRF trigger. The hardware changes necessary to permanently rectify this issue are given in Section 5.4. A plot of three chirps demonstrating the variation in latency using the FPGA software trigger is shown in Figure 5.3.

5.3 Clock Incoherence Between FPGA and DDS System Clock

To further tighten the timing of the pulses, the cause of the 7 ns variation must be determined. Using similar reasoning to that in Section 5.2, the 7 ns variation is

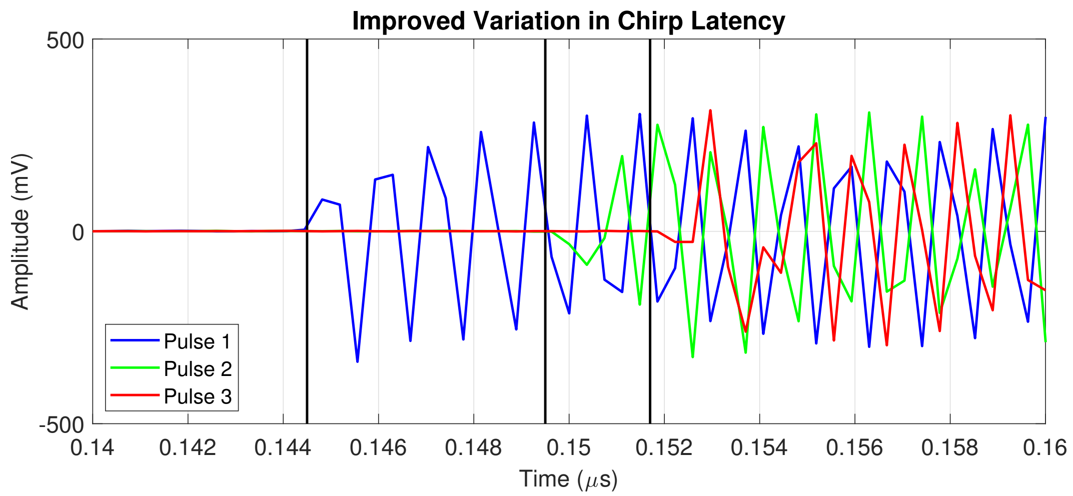


Figure 5.3: Three pulses plotted in time showing close to 7 ns of variation

likely caused by clock edge misalignment involving a clock with a period of 7 ns. The frequency of a clock with a 7 ns period is 142.9 MHz. Although no such clock exists within the system, the DDS timer clock, which is $\frac{1}{24}$ of the 3.5 GHz reconstruction frequency and is given by (4.6), has a clock frequency of 145.83 MHz and a period of 6.86 ns. Given that this clock period lies within predicted range of 7 ± 0.625 ns, the DDS timer clock is likely the culprit. Because the timer clock is used to update the frequency and amplitude output of the DDS, any changes to the waveform (including turning it on and off) must be performed on the edge of the timer clock.

To evaluate why this clock might be an issue, consider the relationship between the FPGA-generated trigger and the DDS system clock. The trigger will be generated on the edge of the 10 MHz LMFC. Although the LMFC and the DDS timer clock are derived from the same 100 MHz oscillator, the process of clock division and multiplication that each one experiences ensures that they are not phase locked. Moreover, they are not integer multiples of one another, ensuring that their clock edges will never consistently align. Given this restriction, the trigger issued from the FPGA will almost certainly alter position with respect to the rising edge of the timer clock, leading to variation in when the DDS output can be turned on.

Because the clocks are derived through PLLs from the same oscillator, they are technically phase-locked to one another. This implies that the coherence issue can be solved by modifying one clock such that the DDS timer clock is an integer multiple of the LMFC. Due to constraints placed on the LMFC by the ADC sampling frequency and the pre-defined values of the JESD204B configuration variables, the LMFC can only be altered by factors of 2. As a result, the DDS timing clock frequency f_{timer}, and therefore the reconstruction clock frequency f_{DDS}, must be modified.

To determine what frequency is acceptable for f_{DDS}, the integer multiple relationship between f_{timer} and f_{LMFC} is described by

$$f_{\text{timer}} = \frac{f_{\text{DDS}}}{24} = k \times f_{\text{LMFC}} \qquad k \in \mathbb{N} \ . \tag{5.1}$$

Solving for f_{DDS} given $f_{\text{LMFC}} = 10$ MHz determines that f_{DDS} must be an integer multiple of 240 MHz.

An additional constraint on f_{DDS} is that the maximum output frequency that can be synthesized is 0.4 times f_{DDS}. Because the maximum output frequency f_{HI} is 1075 MHz, f_{DDS} must be greater than or equal to 2.6875 GHz. Based on these criteria, the list of possible DDS clock frequencies is 2.88 GHz, 3.12 GHz, and 3.36 GHz. Obviously, the highest frequency is desirable as it will produce the highest-fidelity waveform.

To verify that the LMFC can by synchronized on a pulse-to-pulse basis with the DDS timer clock, the DDS frequency f_{DDS} is modified to be 3.36 GHz, placing the timer clock frequency f_{timer} at 140 MHz. This is done by altering the multiplication factor on the PLL to x33.6 rather than x35, which is possible since the ADF4351 has fractional-N multiplication capabilities. The registers in the DDS relevant to the frequency ramp are modified as well using (4.3)-(4.8) to recompute the register values for the same f_{HI}, f_{LO}, and T_p, but with $f_{\text{DDS}} = 3.36$ GHz. The results of multiple pulses transmitted through loopback using the FPGA trigger and the new DDS frequency are shown in Figure 5.4. In this plot, it can be seen that the three pulses are indistiguishable from one another, implying that the latency variation has been eliminated. The timing variation was measured again using a 40 GSPS oscilloscope and determined to be less than 60 ps, resulting in a range ambiguity of 1 cm. This significantly lowered timing variation and range ambiguity are deemed

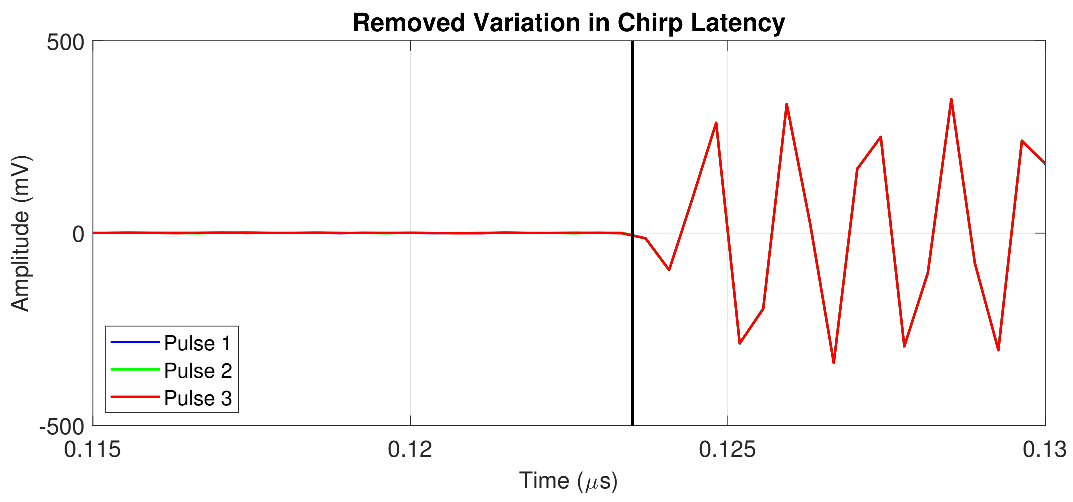


Figure 5.4: Three pulses plotted in time showing lack of variation

to be acceptable as the clock incoherence issue is apparently resolved.

5.4 System Modifications

Based on the conclusions of Sections 5.2 and 5.3, the cause of the timing variation has been narrowed down, but the fixes used to eliminate the variation are not useful for a permanent solution. To obtain optimal performance, the system must be implemented such that the pulses are transmitted coherently and with the correct frequency ramp and amplitude window applied. Comparing the FPGA triggering with the microcontroller triggering, these two goals are in opposition with each other. One possible solution is to modify the FPGA firmware to enable triggers that window the pulse to 1 μs, enable an offset for the *IOUPDATE* pin to initiate the DRG at the correct time, and are generated at 3 kHz rather than on the press of a button in HSDC Pro. However, a much simpler solution is to modify a microcontroller's clocking to be coherent with the rest of the system. If the system clock of the mircontroller is derived from the same oscillator as the ADC, FPGA, and DDS clocks, then it will theoretically be capable of generating a PRF signal whose edges align with both the LMFC and the DDS timer clock so long as its final clock frequency is carefully selected. Furthermore, the pulses can be windowed properly using the same PWM windowing technique developed in Chapter 4.

To clock the microcontroller synchronously with the rest of the system, the microcontroller CPU clock must be replaced with a clock derived in some way from the same coherent reference signal employed by the rest of the system. Generally, the only available clock references within appropriate range of a typical microcontroller clock (usually no greater than $\sim$120 MHz) are the 100 MHz ADC oscillator reference and the DDS timer clock, which can be broken off the DDS evaluation board through the *SYNCCLK* SMA port.

At this junction in the system construction, the entire clock distribution scheme is reconsidered from an overall coherence standpoint. Consider the phase of the DDS timer clock f_{timer}. Because it is internally divided down by 24 from the sampling clock f_{DDS}, it can take on 1 of 24 different phases each time the DDS is turned on. Furthermore, while the DDS does have a mechanism for selecting the phase of the timer clock upon system startup, it is not convenient for use in this application given the specific input clock frequency required for synchronization and it is not guaranteed to function properly if f_{DDS} is greater than 2.5 GHz [22]. As a result, there is no way to exactly synchronize the phase of the DDS timer clock with the rest of the system.

To get around this restriction, the clock distribution scheme in the system is redesigned to use f_{timer} as the coherent reference rather than the oscillator onboard the ADC. Furthermore, f_{timer} is set to 100 MHz to force it to be related by an integer value to the f_s of 1.6 GHz. This way, the ADC will be phase-locked with the DDS regardless of the random clock phase selected by the DDS for f_{timer}. To implement this modification in parallel with the coherent microcontroller clocking, several changes must be made to the rest of the back-end. These are described in Section 5.4.1. Additionally, to configure f_{timer} to 100 MHz, f_{DDS} is altered to 2.4 GHz, requiring modifications to the signal generation and therefore the Tx IF portion of the RF front-end. These are described in Section 5.4.2.

5.4.1 Digital Back-End Redesign

To avoid power-splitting the clock signal out of the 50 Ω *SYNCCLK* SMA port, a scheme is derived where the generated clocks are "daisy-chained" between devices. Additionally, this clock scheme was derived in an attempt to eliminate timing inconsistency between system resets; however, this ultimately proved unsuccessful. The solutions for this reset coherency problem are discussed in Chapter 6. A simplified diagram of the proposed clock distribution scheme is shown in Figure 5.5. In this scheme, the DDS timer clock is used as the base clock to the microcontroller. The microcontroller then uses its PWM module to divide f_{timer} down to a lower frequency reference. This reference is then passed to the ADC in place of its onboard 100 MHz oscillator and the high-frequency clocks are derived from the reference. As before, the 3 kHz PRF is also generated on the microcontroller to trigger the DDS and FPGA.

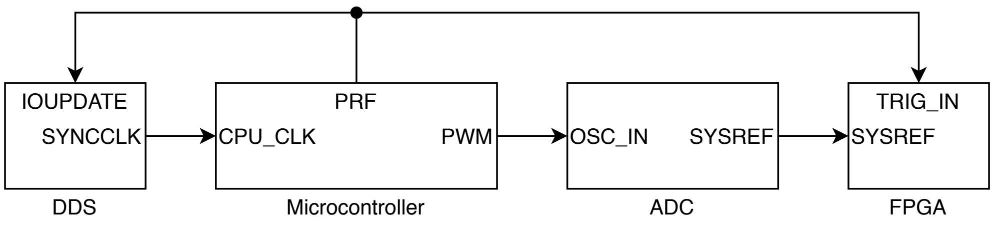


Figure 5.5: A simplified diagram of the proposed new clock distribution scheme derived from the DDS timer clock

In theory, both the microcontroller and ADC could be clocked from the output of *SYNCCLK*. However, when this was attempted, the microcontroller would stop working once the clock driver was transitioned to use the external rather than internal clock reference. Therefore, it was determined that using the PWM module as a clock divider and buffer would allow both the microcontroller and ADC to be clocked from derivative signals of the DDS timer clock.

To implement the new microcontroller clocking solution, a microcontroller is required that can accept a 100+ MHz input clock signal. While the LPC1768 can operate at speeds up to 100 MHz through the use of an internal PLL, the highest frequency it can accept as an external clock input is 50 MHz [21]. Therefore, a new microcontroller that can operate with a single-ended clock input up to 100 MHz is selected - the Texas Instruments LAUNCHXL-F280049C (shown in Figure 5.6) [23]. The layout of this microcontroller development board makes it simple to remove the onboard oscillator and solder leads to the oscillator pins to input the external clock signal to the chip [24].

The first step in enacting the modifications to the digital back-end is to reconfigure the DDS to be operated at a reconstruction frequency f_{DDS} of 2.4 GHz and

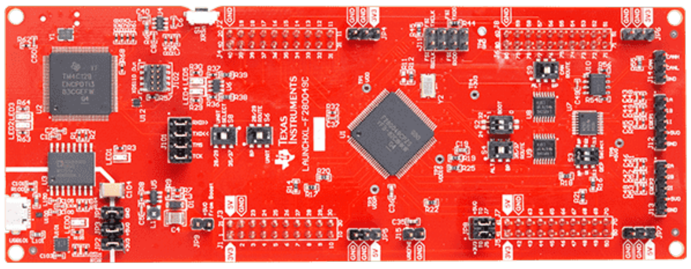


Figure 5.6: The Texas Instruments LAUNCHXL 280049C microcontroller

the timer clock frequency f_{timer} of 100 MHz. This is a convenient clock frequency because the DDS contains an internal PLL that is designed to operate with an output between 2.4 and 2.5 GHz. As a result, the DDS can be clocked by a 3.3 V 100 MHz oscillator connected directly to the *REF IN* SMA port and internally derive the 2.4 GHz clock frequency. The ADF4351 PLL can be removed from the system, simplifying the layout and power distribution of the system as a whole.

A major consequence of reducing the DDS reconstruction frequency to 2.4 GHz is that the maximum output frequency is reduced to 960 MHz, making the generation of an LFM pulse from 925 to 1075 MHz infeasible. Instead, the output frequency is divided by two to give an output sweeping from 462.5 MHz to 537.5 MHz - an LFM pulse with 75 MHz of bandwidth centered at 500 MHz. The RF front-end is then modified to multiply by x4 in the IF section rather than x2, resulting in the same signal being transmitted. These changes to the RF front-end are discussed in Section 5.4.2. Using (4.3)-(4.8), the appropriate register values for the proposed DDS configuration are computed. These values are given in Table 5.1.

The 100 MHz timer clock is broken off the board through the *SYNCCLK* SMA port and input to the microcontroller clock through a twisted pair transmission line. The 20 MHz oscillator on the microcontroller is removed from the board and replaced by soldering the leads of the twisted pair to the oscillator pads, shown in Figure 5.7. By default, the microcontroller is clocked using an internal 10 MHz oscillator (INTOSC2) which is multiplied to 100 MHz by the microcontroller PLL [25]. Switching over to the external oscillator as the clock source requires only changing the clock control register CLKSRCCTL1.OSCCLKSRCSEL from 0x0 to 0x1 and the the register XTALCR.SE from 0x0 to 0x1 to enable a single-ended clock input.

In order to daisy-chain the clock from the microcontroller to the ADC, the PWM

Parameter	Register Name	Parameter Value	Register Value (Hex)
f_{LO}	FTW_{LO}	462.5 MHz	0x31555555
f_{HI}	FTW_{HI}	537.5 MHz	0x39555555
Δf	STEP$_P$	0.75 MHz	0x00147AE1
Δt	P	10 ns	0x0001

Table 5.1: The register values used to configure the DDS DRG when the DDS is clocked at 2.4 GHz

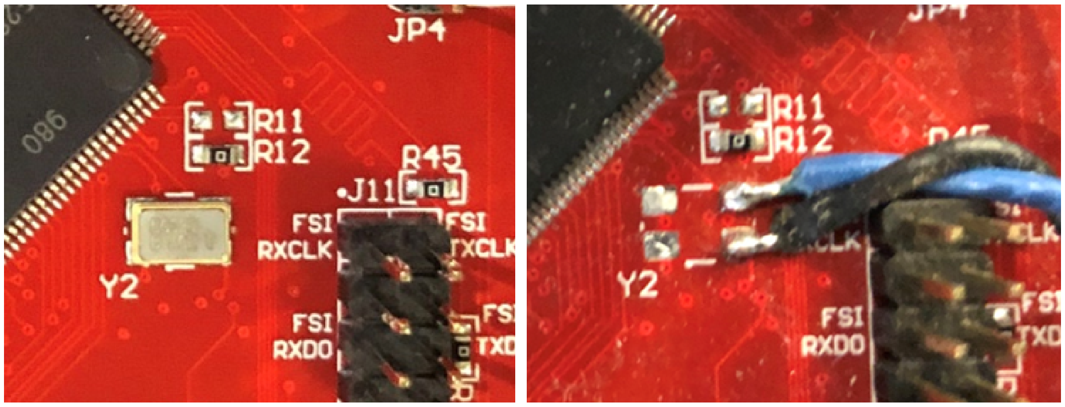


Figure 5.7: The microcontroller oscillator before (left) and after (right) the twisted pair carrying the DDS timer clock reference is soldered to the pads

subsystem is used to generate a 25 MHz clock reference. This can be generated along with the 3 kHz PRF triggers since the LAUNCHXL-F280049C has 8 distinct PWM modules, each containing two PWM channels and with each being capable of operating with a different period [25]. The PWM is setup easily by setting *MR* to 3, *MR1* to 1, and *MR2* to 3, generating a clock output of 25 MHz with a duty cycle of 50%.

Connecting the 25 MHz clock as the reference to the ADC requires a few additional steps. First, the 100 MHz oscillator must be disconnected from the TRF3765. A 0 Ω resistor (R38) connects the output of the oscillator to both the *OSC IN* SMA port (from which the reference was originally obtained) and the clock input to the TRF3765. Removing this resistor disconnects the oscillator and allows the 25 MHz PWM signal to be input to *OSC IN*.

Another modification is to reconfigure the TRF3765 to multiply the 25 MHz signal up to 1.6 GHz for the ADC sampling clock. The TRF3765 is a PLL with a very similar architecture to that of the ADF4351 shown in Figure 4.1. The only relevant differences are that there are not x2 and /2 blocks immediately after the REF_{IN} input, and the N divider in the feedback loop is split into two dividers - an N divider which can take on any value from 1 to 65535, and an N_{RF} divider which can only take on the values of 1, 2, or 4. The default parameters of the device are $R = 2$, $N = 16$, $N_{RF} = 4$, and $D_{RF} = 2$. For an input REF_{IN} of 100 MHz, this gives an f_{PFD} of 50 MHz, an f_{VCO} of 3.2 GHz, and an output RF_{OUT} of 1.6 GHz. To accomodate the new REF_{IN} of 25 MHz, the TRF3765 configuration file is modified so that $N = 64$. This change causes f_{PFD} to be 12.5 MHz but leaves the output at 1.6 GHz.

The final change to the back-end, which was implemented in an unsuccessful attempt to synchronize the DDS phase with the LMFC on system startup, is to change the LMFC from 10 MHz to 2.5 MHz. Despite the change not producing the desired effect, it was kept in the system because it forces the LMFC to be an integer multiple of the 25 MHz reference produced by the microcontroller from which the LMFC is derived. Implementing a 2.5 MHz LMFC requires both that the SYSREF signal generated by the LMK04828 be set to 2.5 MHz and that the JESD204B K value be modified. Solving (2.15) for K gives a new K value of 16.

Modifying the SYSREF signal on the LMK04828 is straightforward. A register on the device (SYSREF_DIV) specifies the amount to divide the input clock to obtain the SYSREF signal. Since the input clock is half the sampling frequency, or 800 MHz, the default value of the register is 80. Modifying the register to a value of 320 divides the output to 2.5 MHz.

Changing the K value is also straightforward. To do so on the ADC requires that the K-minus-1 (KM1) register be changed from 3 to 15. Changing the K value on the FPGA is a little more complicated because the JESD204B parameters in the FPGA configuration file are not defined correctly - the default value of K is listed as 32, and the default value of F is listed as 1, neither of which is the correct value listed in Table 4.2. According to a TI engineer in [26], the K value in the configuration file should be set to the product of K and F. Therefore, to set K to 16, the value of K in the configuration file should be set to 128.

The longer period of the LMFC must be considered when setting up the PWM signals for the PRF. Because the new LMFC period is 400 ns, the period of the PWM must be an integer multiple of 400 ns, or 40 clock cycles at a clock speed of 100 MHz. This will ensure that every trigger generated by the microcontroller to begin the waveform generation and capture is synchronous with the LMFC, preventing any latency variation between pulses. The value of *MR* that meets this criterion and gives the PWM period closest to 333.3 μs (3 kHz) is 33199. Setting *MR1* to 33099 and *MR2* to 33199 gives a period of 100 clock cycles (1 μs) for the *OSK* window. The *IOUPDATE* trigger is generated using an *MR1* value of 33095, allowing it to lead the amplitude window to the DDS by 40 ns.

It is possible that the PRF will be stopped and restarted during the operation of the radar. As such, it is desirable that the PWM module being used to generate the PRF signals have some method of synchronizing the phase of the PWM timer with the edge of the LMFC. To do this, a minor firmware modification was solicited from TI for the FPGA, which allows the LMFC signal internal to the FPGA to be broken out on the *TRIG OUT B* SMA connector [27]. This signal is connected to a GPIO pin of the microcontroller that is multiplexed to synchronize the timebase of the microcontroller with a rising edge whenever the register TBCTL.PHSEN is set to 0x1. As such, when the PWM PRF is turned on, the TBCTL.PHSET bits for the relevant PWM modules are enabled for 1.2 μs to guarantee that at least one clock edge of the LMFC will synchronize the phase of the PWM module. This solution is tested and it is shown that turning the PRF off and back on does not modify the timing between the transmission and receipt of a pulse.

A detailed diagram of the clock distribution scheme described in this section is shown in Figure 5.8. With the implementation of these changes, the digital system is expected to maintain coherency. When tested in loopback with the output of the DDS connected directly to the input of the ADC, the sequence of pulses appears similar to that in Figure 5.4. The spectrum and instantaneous frequency plot are expected to appear similar to those shown in Figure 4.10, but with a frequency range from 462.5 MHz to 537.5 MHz rather than 925 MHz to 1075 MHz. The final signal characterisitics including the time-domain signal, spectrum, and instantaneous frequency plot are shown in Figure 5.9. These pulse characteristics confirm that the modifications to the digital back-end have not compromised the desired generation of the pulse.

5.4.2 RF Front-End Modifications

To accomodate the change in output frequency from the DDS, a few modifications need to be made to the RF front-end. The waveform generated by the digital back-end is identical to the original, but with half the bandwidth and half the center frequency. As such, the analog front-end can basically be left unchanged but with an additional x2 frequency multiplier placed right after the output of the DDS. Theoretically, a x4 multiplier could be used in place of two x2 multipliers. In practice, however, the unwanted output harmonics spaced only 500 MHz apart would be very difficult to reliably suppress with a practical bandpass filter.

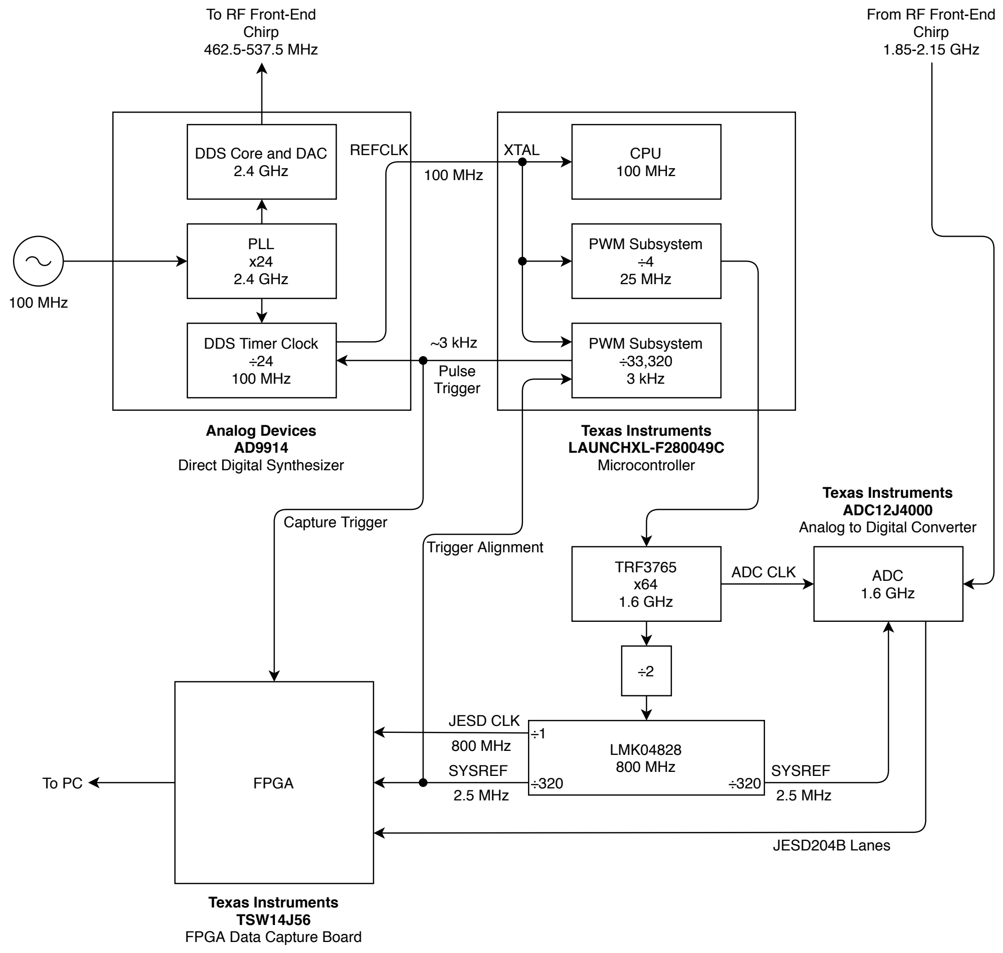


Figure 5.8: The clock distribution scheme in the updated digital back-end

Consider the IF chain on the transmitter of the radar, shown in Figure 5.10. It accepts a 1 GHz LFM with 150 MHz of bandwidth from the DDS with a power of -1.5 dBm, and outputs a 2 GHz LFM with 300 MHz of bandwidth to the mixer at a power of -3.5 dBm. The new IF will accept a 500 MHz LFM with 75 MHz of bandwidth, also with a power of -1.5 dBm. To minimize the changes to the front-end, it is desirable to add components between the original IF chain and the DDS to condition the new input waveform to match the original waveform in frequency and be as close as possible in power level.

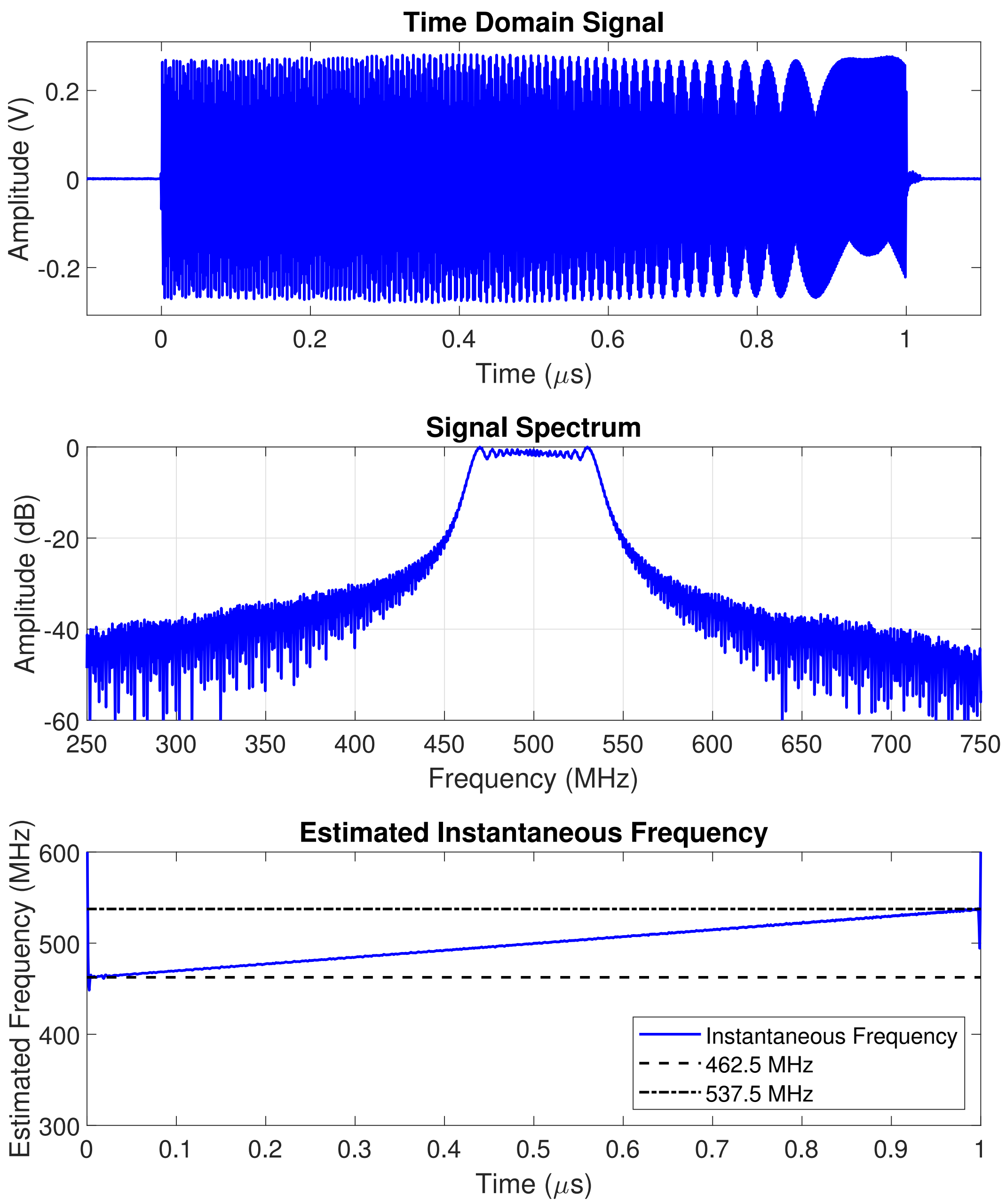
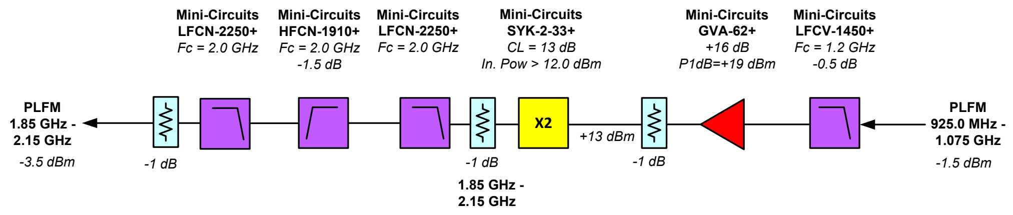


Figure 5.9: The time-domain pulse (top), spectral content (middle), and instantaneous frequency (bottom) of the 500 MHz LFM pulse after clock domain modifications to the digital back-end

The first new component in the IF chain is a lowpass filter with a cutoff near the 537.5 MHz high frequency output of the DDS. The filter that is selected is the Mini-Circuits LFCN-575+, which has a 3-dB cutoff frequency of 770 MHz and an



Mini-Circuits LFCN-2250+ Fc = 2.0 GHz
Mini-Circuits HFCN-1910+ Fc = 2.0 GHz -1.5 dB
Mini-Circuits LFCN-2250+ Fc = 2.0 GHz
Mini-Circuits SYK-2-33+ CL = 13 dB In. Pow > 12.0 dBm
Mini-Circuits GVA-62+ +16 dB P1dB=+19 dBm
Mini-Circuits LFCV-1450+ Fc = 1.2 GHz -0.5 dB
PLFM 1.85 GHz - 2.15 GHz
-3.5 dBm
-1 dB
-1 dB
1.85 GHz - 2.15 GHz
X2
+13 dBm
-1 dB
PLFM 925.0 MHz - 1.075 GHz
-1.5 dBm

Figure 5.10: The original IF chain on the radar transmitter

insertion loss of 0.75 dB in the signal band [28]. The filter has a rejection of over 35 dB at $\frac{f_s}{2}$ (1.2 GHz), which is sufficient to eliminate the image signals in higher Nyquist zones.

The x2 multiplier that is selected is the Mini-Circuits SYK-2R+ because it has an input frequency range of 10 MHz to 1000 MHz [29]. It requires an input power level between 12 dBm and 16 dBm, and has a typical conversion loss of 11 dB. The signal power at the output of the lowpass filter is -2.25 dBm. To obtain the drive power required by the SYK-2R+, this signal must be amplified by between 14.25 dB and 18.25 dB. The amplifier selected for this is the Mini-Circuits GVA-62+ because is has an acceptable gain of 15.75 dB in the signal band and it is used elsewhere in the system, reducing the number of spare parts required to keep on hand [30].

The output of the x2 multiplier is the 1 GHz LFM pulse at a nominal power level of 2.5 dBm. It must be filtered to eliminate the potentially large 4th order harmonic output at 2 GHz, which is only 10 dB lower than the desired output signal [29]. In addition to the original reconstruction lowpass filter, the Mini-Circuits LFCV-1450+, the Mini-Circuits CBP-1000F+ bandpass filter is included at the output of the multiplier. This filter has high selectivity between 900 and 1100 MHz, contributing to the suppression of the spurs at the output of the x2 multiplier [31]. Including the rejection by both filters and the harmonic suppression provided by the x2 multiplier, the 1st-, 3rd-, and 4th-order harmonics are all suppressed by more than 100 dBc relative to the output of the desired 2nd-harmonic. To force the final output power to be close to the original power level, a 4 dB attenuator is included between the 2x multiplier and the cascaded filters, resulting in a final output power of -4 dBm provided to the mixers, which deviates from the original output power by only 0.5 dBm. The final IF chain block diagram is shown in Figure 5.11.

The signal spectrum after a full RF loopback test is shown in Figure 5.12 and appears nearly identical to the original in Figure 4.11, indicating that the new IF chain does not negatively impact RF performance.

5.5 Summary

In this section, multiple pulse-to-pulse timing incoherence prolems were discovered and addressed. The solutions required some significant modifications to the clock distribution scheme in the system and a slight increase in complexity of the RF front-end, but the fundamental digital hardware was not replaced and the

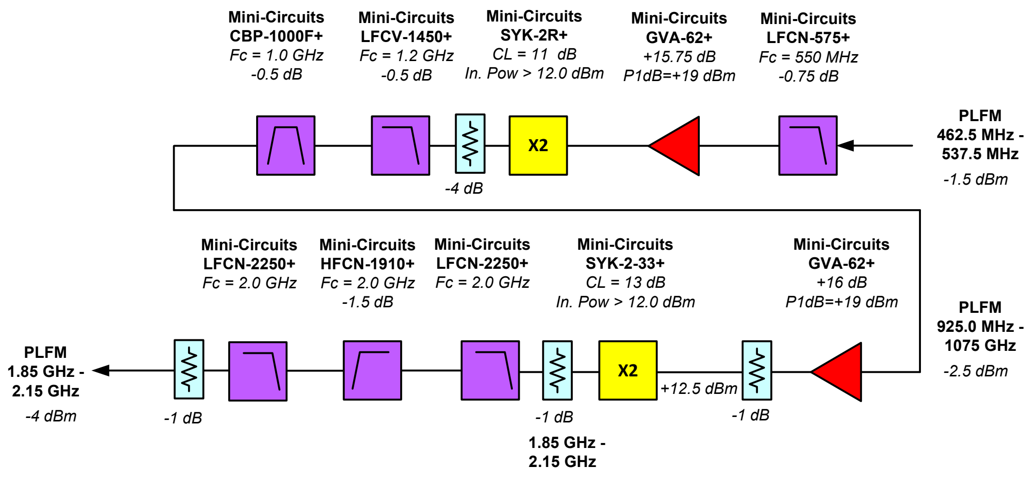


Figure 5.11: The updated IF chain on the radar transmitter

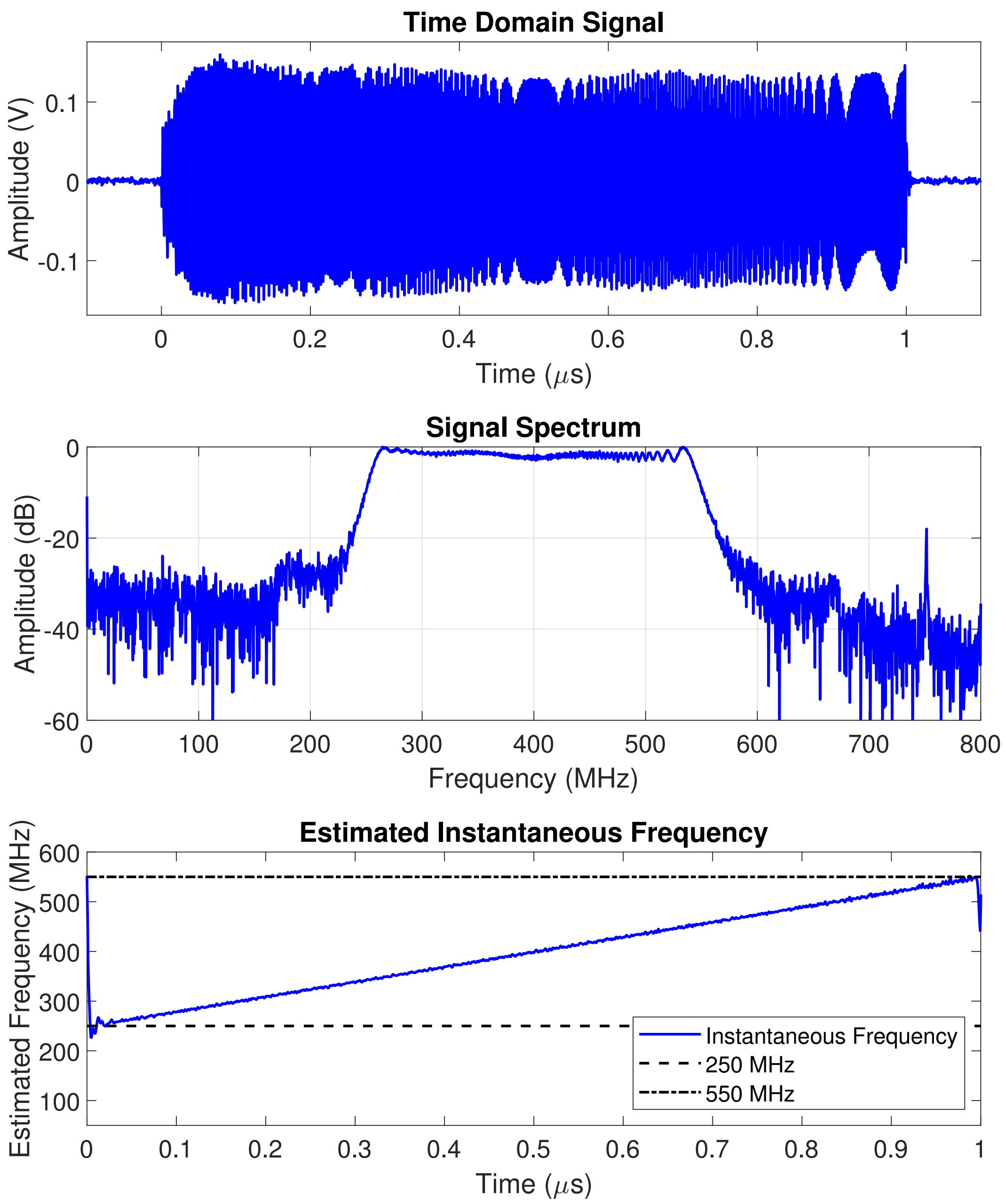


Figure 5.12: The time-domain pulse (top), spectral content (middle), and instantaneous frequency (bottom) of the LFM pulse through the RF front-end with the updated IF section

system parameters and performance have not been altered. Furthermore, the modifications allow the radar to transmit coherent pulses at the desired PRF with little to no timing jitter without sacrificing the ideal spectral content of the LFM pulse.

Chapter 6

Start-Up Incoherence and Loopback Calibration

In Chapter 5, a pulse-to-pulse incoherence that caused each pulse to have a variable latency between the digital transmitter and receiver was discovered and resolved. Resolving this issue ensures that pulses can integrate coherently and, when combined over an entire CPI, perform range-Doppler processing as seen in Figure 2.6 rather than the unfocused one shown in Figure 2.8. However, in order to accurately plot range data, the two way propagation time τ_p must be known. Based on (2.12), the values of τ_{RF} and τ_{dig} must be known and calibrated out along with T_p for τ_p to be extracted from τ_m.

During system testing, it is determined that despite the pulse-to-pulse timing being resolved, the latency between the DDS and ADC changes if the system is turned off and back on again. Although the pulses are still timed identically from pulse to pulse and will integrate coherently, the range axis cannot be reliably set because τ_{dig} is not known consistently. In this chapter, solutions to this problem are proposed. Some modifications to the digital clocking scheme that ultimately proved unsuccessful are briefly discussed. The solution that fixed the issue is discussed in detail and test data proving the efficacy of the implemented solution is presented.

6.1 Initial System Modification Attempt

The ideal solution will completely eliminate the latency variation between system resets such that τ_{dig} remains a constant whenever the system is turned on. The amount of time that the latency would vary by is close to the period of the LMFC (100 ns at one point, 400 ns at another point), indicating that the LMFC phase is not being set up consistently with respect to some other important clock domain.

Two different approaches were taken to fix the issue by deriving a new clocking scheme. The first attempted solution was to bypass the LMK04828 on the ADC evaluation board and instead generate a 10 MHz SYSREF signal on the microcontroller. Theoretically, this would allow the phase of the SYSREF signal to be controlled and give a guarantee that it would be lined up with the edge of the PRF trigger. Although the new SYSREF source did not cause additional problems, it also did not reduce the timing variation when the system was reset.

The second approach is the original reason that the LMFC frequency was modified to 2.5 MHz in Chapter 5. As mentioned previously, the DDS contains an synchronization input/output (*SYNC IN* and *SYNC OUT*) that is used to synchronize the phases of the DDS timer clocks across multiple DDS devices in a multichannel system. The synchronization input/output operates at $\frac{1}{16}$ the frequency of f_{timer}. If the DDS reconstruction frequency f_{DDS} is set to 2.88 GHz, f_{timer} will be 120 MHz and the synchronization signal will be at 7.5 MHz, which is an integer multiple of 2.5 MHz. Since 2.5 MHz is attainable as an LMFC frequency, it could be multiplied at the output of the FPGA by a low frequency PLL to obtain 7.5 MHz and then used to synchronize the phase of the DDS with the FPGA. Because the PRF trigger can also be synchronized to the phase of the LMFC, this would theoretically lock all of the clocks together in phase on startup and eliminate the reset-to-reset latency variation.

Unfortunately, this was not successful. This is potentially due to some flawed premise in the setup, but could also be due to the fact that the synchronization input is not guaranteed to work when f_{DDS} is above 2.5 GHz. After multiple attempts to get a clever clocking solution to function, it is decided that a solution other than a large modification to the clocking scheme would be more practical.

6.2 Theory of Loopback Calibration

Rather than trying to eliminate the reset-to-reset incoherence, it is more practical and feasible to determine a method for compensating for it during processing. It is desired to determine what the variation in delay is whenever the system is turned on. To do so, the DDS and ADC must be able to cooperate in capturing a single pulse with a known propagation delay between the DDS and ADC. It is simple to do this using a delay line or using a large reflector with a precisely known distance from the radar. In this way, the sum of τ_{RF} and τ_{dig} can be determined by rearranging (2.12) because τ_p, τ_m, and T_p are known.

However, calibrating the radar like this by using external equipment or a precisely located reflector is not practical if the calibration must be performed each time the radar is turned on. Moreover, since the RF front-end can be expected to have a reliably constant group delay, the calibration only really needs to be performed directly between the DDS and ADC. To that end, it is proposed to include RF switches directly at the output of the DDS and the input of the ADC with options to multiplex the signal either through the RF front-end or through a direct cable connecting the switches. A basic diagram of this setup is shown in Figure 6.1.

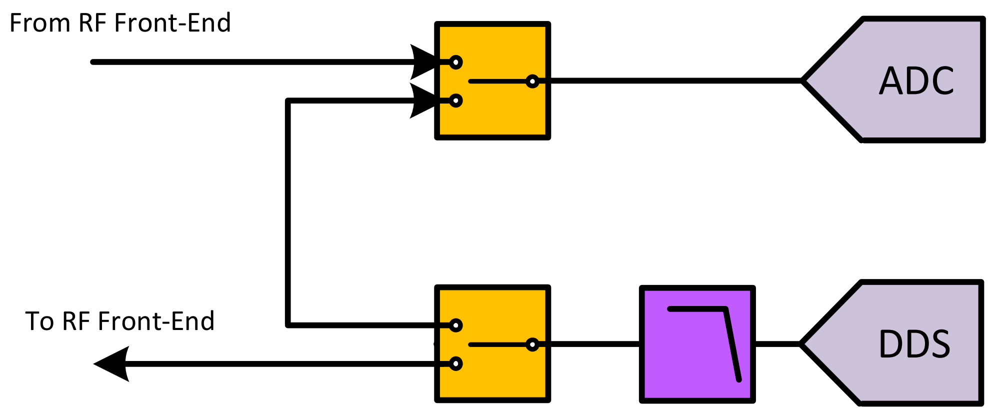


Figure 6.1: The proposed RF switch digital loopback setup

6.2.1 Derivation

With the concept for the loopback calibration established, the methodology for implementing it must be established. To begin, there are multiple different sources of propagation delay within the system that contribute to the round-trip delay of a signal from generation by the DDS to receipt by the ADC and FPGA. To effectively calibrate the system, these delays must first be disambiguated.

The most important delay is that caused by the actual wave propagation to a measurement scene, τ_p. For the purposes of radar calibration, we refer to this as τ_{cal} since the radar will utilize a calibration cable whose delay is known. The delays introduced by the RF front-end transmitter and receiver are denoted by τ_{Tx} and τ_{Rx}, respectively. The delay through each RF switch is denoted by τ_{sw}. Note that this derivation assumes that the delay through each switch is identical, but the end result does not depend on this being true. The loopback cable delay between the two switches is denoted by τ_{loop}. Finally, the variable delay between the DDS and ADC is given by τ_{dig}. The signal transmitted through the RF front-end and the calibration cable will be delayed by

$$\tau_{\text{m,cal}} = 2\tau_{\text{sw}} + \tau_{\text{Tx}} + \tau_{\text{cal}} + \tau_{\text{Rx}} + \tau_{\text{dig}} \tag{6.1}$$

while the signal transmitted through the switch loopback will be delayed by

$$\tau_{\text{m,loop}} = 2\tau_{\text{sw}} + \tau_{\text{loop}} + \tau_{\text{dig}} \quad . \tag{6.2}$$

A high-level diagram of the system with the different delay sources labelled is shown in Figure 6.2.

Consider the real-valued continuous-time signals $r_0(t)$ and $d_0(t)$. These are the analog radar waveform as it is received by the ADC and as it is generated by the DDS, respectively, with the waveform starting at $t = 0$ and ending at $t = T_p$. Assume that the system is turned on and a pulse is transmitted through the RF

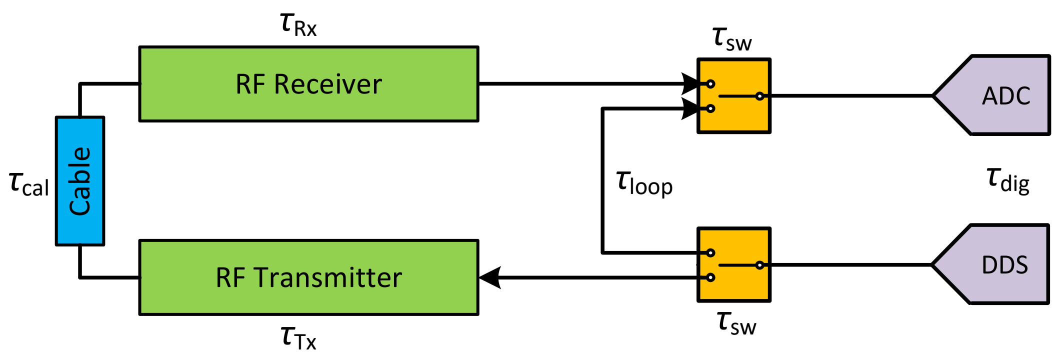


Figure 6.2: A diagram showing the different delays in the radar system

front-end and the calibration cable, giving an RF calibration signal

$$r_{\text{cal}}(t) = r_0(t - \tau_{\text{m,cal}}) \tag{6.3}$$

and a pulse is transmitted through the loopback path, giving a digital calibration signal

$$d_{\text{cal}}(t) = d_0(t - \tau_{\text{m,loop}}) \ . \tag{6.4}$$

Recall that the cross-correlation of two functions $f_1(t)$ and $f_2(t)$ given by

$$R_{f_1 f_2}(\tau) = \int_{-\infty}^{\infty} f_1(t) f_2(t + \tau) dt \tag{6.5}$$

will have a maximum at the lag time at which $f_2(t)$ is most similar to $f_1(t)$. Therefore, the cross-correlation between $r_0(t)$ and $r_{\text{cal}}(t)$ given by $R_{rr}(\tau)$ will have a maximum at $\tau = \tau_{\text{m,cal}}$. Backing out τ_{cal} gives

$$\tau_{\text{m,cal}} - \tau_{\text{cal}} = 2\tau_{\text{sw}} + \tau_{\text{Tx}} + \tau_{\text{Rx}} + \tau_{\text{dig}} \equiv \tau_0 \tag{6.6}$$

where τ_0 is defined as the delay introduced by the system during the calibration. Similarly, the cross-correlation between $d_0(t)$ and $d_{\text{cal}}(t)$ given by $R_{dd}(\tau)$ will have a maximum at $\tau = \tau_{\text{m,loop}}$. The value of $\tau_{\text{m,loop}}$ obtained during the calibration routine is defined as τ_{d0}. Because τ_{cal} is the value of interest in the radar measurement, the time axis used to plot the matched filtered signal should be adjusted by subtracting τ_0, or the range axis can be adjusted directly by subtracting $\frac{c\tau_0}{2}$. The value of τ_0 and the sequence $d_{\text{cal}}(t)$ are saved for use in calibrating the radar on startup later.

To begin ordinary radar operation, the system is turned off and the calibration cable is disconnected from the radar. The radar front-end is connected to antennae through cables which collectively introduce a known delay τ_{ant}, shown with the rest of the system delays in Figure 6.3. This eliminates the possibility of collecting a reliably delayed signal through the RF front-end; however, a signal can still be transmitted through the loopback switches. This signal is denoted as $d_1(t)$. With the assumption that τ_{sw} and τ_{loop} have not changed but that the digital delay has changed by some amount $\Delta\tau_{\text{dig}}$, the cross-correlation between $d_0(t)$ and $d_1(t)$ will produce a maximum at $\tau_{d0} + \Delta\tau_{\text{dig}}$. More directly, the cross-correlation of $d_{\text{cal}}(t)$ with $d_1(t)$ will have a maximum at $\Delta\tau_{\text{dig}}$. Because τ_{Tx}, τ_{Rx}, and τ_{ant} are all assumed to stay constant as well, the time axis for the current system runtime can be adjusted by subtracting the sum of $\tau_0 + \Delta\tau_{\text{dig}} + \tau_{\text{ant}}$.

Similarly to (2.8), note that the cross-correlation of two signals $f_1(t)$ and $f_2(t)$ can be rewritten by

$$\begin{aligned} R_{f_1 f_2}(\tau) &= \int_{-\infty}^{\infty} f_1(t) f_2(t+\tau)dt \\ &= \int_{-\infty}^{\infty} \hat{f}_1(\tau - t) f_2(t)dt \\ &= \hat{f}_1(t) * f_2(t) \end{aligned} \tag{6.7}$$

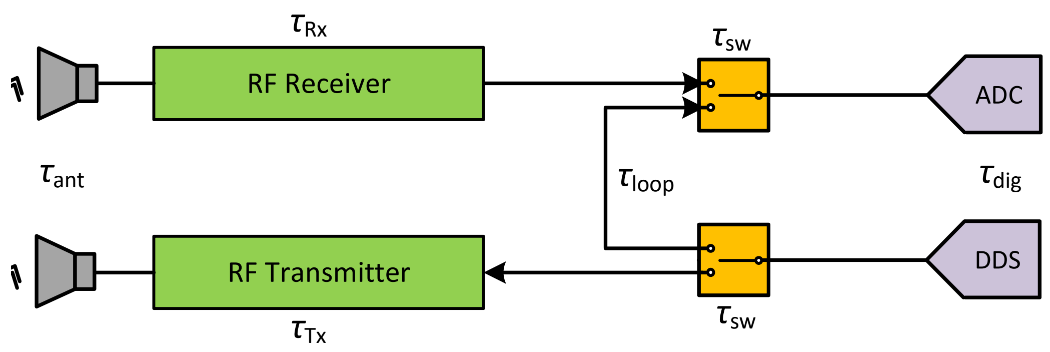


Figure 6.3: A diagram showing the delays in the system with the antenna

where

$$\hat{f}_1(t) = f_1(-t) \quad . \tag{6.8}$$

This shows that the cross-correlation can be implemented as a convolution with a time-reversed $f_1(t)$. Furthermore, properties of the Fourier Transform make it convenient to express the convolution operation as a multiplication in the frequency domain. That is,

$$S_{f_1 f_2}(\omega) = \hat{F}_1(\omega) F_2(\omega) \tag{6.9}$$

where

$$\begin{aligned} \hat{F}_1(\omega) &= \mathcal{F}\left\{\hat{f}_1(t)\right\} \\ F_2(\omega) &= \mathcal{F}\left\{f_2(t)\right\} \\ S_{f_1 f_2}(\omega) &= \mathcal{F}\left\{R_{f_1 f_2}(\tau)\right\} \quad . \end{aligned} \tag{6.10}$$

Thus,

$$\begin{aligned} R_{rr} &= \mathcal{F}^{-1}\left\{\hat{R}_0(\omega) R_{\text{cal}}(\omega)\right\} \\ R_{dd} &= \mathcal{F}^{-1}\left\{\hat{D}_{\text{cal}}(\omega) D_1(\omega)\right\} \quad . \end{aligned} \tag{6.11}$$

This property will be convenient when the cross-correlation is implemented digitally.

6.2.2 Digital Algorithm Implementation

The above derivation in continous time shows how the variation in τ_{dig} can be characterized each time the radar is turned on. However, the method must be defined carefully when it is implemented using the real digital hardware comprising the radar back-end. Consider the N-point digital sequences $d_0[n]$ and $r_0[n]$ where

$$N = f_s T_p + 1 = 1601 \tag{6.12}$$

and the M-point digital sequences $r_{\text{cal}}[n]$, $d_{\text{cal}}[n]$, and $d_1[n]$ where M is an arbitrary number of captured samples. In the actual implementation in MATLAB, M is set to 4096. All of the above sequences are discrete-time representations of the continuous-time signals defined previously.

To find τ_0, a similar approach to the continuous-time version is taken, exploiting the Fourier Transform property of cross-correlation defined in (6.11). First, the time-reversed signal $\hat{r}_0[n]$ is created by flipping the order of the sequence. Next, the K-point discrete Fourier Transform (DFT) of $\hat{r}_0[n]$ and $r_{\text{cal}}[n]$ are both taken, where

$$K = M + N - 1 \tag{6.13}$$

which is accomplished using the fft command in MATLAB. The DFT sequences are multiplied together, resulting in $S_{rr}[k]$ where $k \in [0, K-1]$ indexes frequencies in $[0, \frac{K-1}{K} f_s]$. The frequency domain representation of the cross-correlation is useful because the DFT can be zero-padded to a length of αK, and when the inverse DFT is applied, the resulting sequence will have a time-domain resolution that is α times finer than the original sequences. The value of 16 is selected for α in this application. This will give a more precise time value associated with the maximum in $R_{rr}[n]$.

After zero-padding to a length αK, the inverse DFT is taken using the ifft command in MATLAB to obtain an αK length sequence for $R_{rr}[n]$. Because $r_0[n]$ was originally indexed on $[0, N-1]$, the time-reversed sequence $\hat{r}_0[n]$ is indexed on

$[-N+1, 0]$. The calibration sequence $r_{\text{cal}}[n]$ was indexed on $[0, M-1]$. The time values associated with the sequence $R_{rr}[n]$ will therefore be on $[-T_p, \frac{\alpha M-1}{\alpha f_s}]$.

The assumption is made that with a short cable loopback, the SNR will be very high. This guarantees that the peak of $R_{rr}[n]$ will correspond to the time point $\tau_{\text{m,cal}}$ and not another peak caused by noise. Therefore, the value of time associated with the index of $\max\left(R_{rr}[n]\right)$ is a precise estimation of $\tau_{\text{m,cal}}$. Subtracting the known τ_{cal} from $\tau_{\text{m,cal}}$ gives τ_0, which is saved along with a digital loopback sequence $d_{\text{cal}}[n]$ for use later.

Once the system is connected to the antenna as in Figure 6.3 and restarted, the digital loopback sequence $d_1[n]$ is captured. Using the same DFT methodology, the cross-correlation of $d_1[n]$ and $d_{\text{cal}}[n]$, $R_{dd}[n]$, is found. The only difference will be that

$$K = 2M - 1 \tag{6.14}$$

because both $d_1[n]$ and $d_{\text{cal}}[n]$ are length M. Time-reversing $d_{\text{cal}}[n]$ to $\hat{d}_{\text{cal}}[n]$ and taking the K-point DFT of both $\hat{d}_{\text{cal}}[n]$ and $d_1[n]$ and multiplying results in $S_{dd}[k]$. Again zero-padding the DFT to length αK and taking the inverse DFT gives the sequence $R_{dd}[n]$ indexed in time on $\left[-\frac{M}{f_s}, \frac{M-1}{f_s}\right]$. Searching for $\max\left(R_{dd}[n]\right)$ and finding the associated value in time gives $\Delta\tau_{\text{dig}}$. Taking into account the base delay τ_0, the antenna cable delay τ_{ant}, and the change in digital delay $\Delta\tau_{\text{dig}}$, the range axis can be set after pulse compression by

$$R = \frac{c(\tau_m - \tau_0 - \tau_{\text{ant}} - \Delta\tau_{\text{dig}} - T_p)}{2} \tag{6.15}$$

which will correctly align the range profiles to the calibration standard.

6.3 Loopback Calibration Hardware

It is desirable to use the same switch model for both sides of the loopback for simplicity. Because the switch on the ADC must pass both the 500 MHz signal and the 2 GHz signal, it should have a bandwidth from at least 462.5 MHz to 2.15 GHz. Additionally, to avoid reflecting power back at an IF amplifier, the DDS, or ADC, the switches should be non-reflective and terminate into a 50 Ω load when not enabled.

The switch that is selected is the Mini-Circuits HSWA2-63DR+. This switch has an operational bandwidth from 100 MHz to 6 GHz and an insertion loss of 1 dB in the signal bands of interest [32]. The loss can be compensated for by modifying attenuators in the RF front-end to ensure that the signal power at the input of the ADC and the output of the Tx antenna is not changed. The switches have three RF ports *RF COM*, *RF1*, and *RF2*, and two digital control signals *Control 1* and *Control 2*. When both control signals are driven low, any power input to the switch through any port is dissipated internally in a 50 Ω load. When *Control 1* is driven high and *Control 2* is driven low, power can flow in either direction between *RF COM* and *RF1*. Conversely, when *Control 2* is driven high and *Control 1* is driven low, power can flow in either direction between *RF COM* and *RF2*. The switch documentation does not define the switch behavior when both control inputs are driven high.

The switches are connected as shown in Figure 6.4. The control signals are driven using GPIO pins on the LAUNCHXL microcontroller. By default, they are configured to both be driven low so that no power can flow, giving the user the chance to turn on all the equipment before accidentally applying RF power to a device. To enable the loopback calibration path, *Control 1* is driven high on the Tx switch and *Control 2* is driven high on the Rx switch. To enable the normal operation path, *Control 2* is driven high on the Tx switch and *Control 1* is driven high on the Rx switch.

6.4 Calibration Results

To prove that the calibration routine works, a 1 m cable with 80 dB of attenuation is connected to between the Tx and Rx RF ports of the front-end. This cable and attenuator combination has an electrical length of 1.3 m, giving a τ_{cal} of 4.33 ns. The system is turned on and turned off five times, and during each system turn-on an RF sequence $r_i[n]$ and a loopback sequence $d_i[n]$ is captured, $i \in [1, 5]$. The RF calibration is performed using $r_1[n]$ and $d_1[n]$ as $r_{\text{cal}}[n]$ and $d_{\text{cal}}[n]$. The digital sequences $d_i[n]$ are shown in Figure 6.5, and the RF sequences $r_i[n]$ are shown in Figure 6.6. Notice that the offset between $d_i[n]$ and $r_i[n]$ is the same for all i.

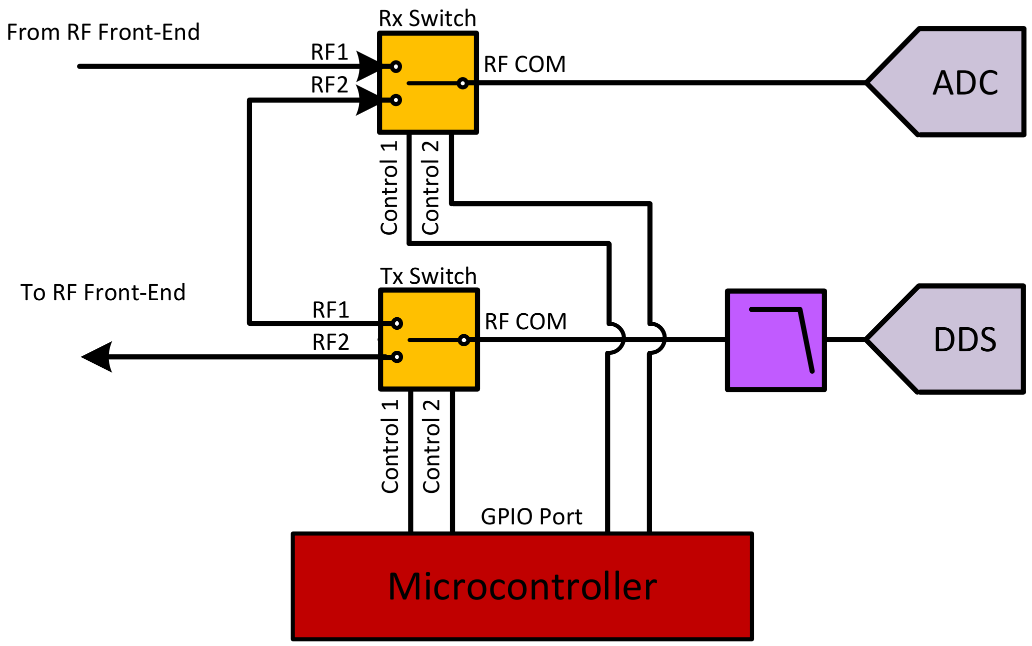


Figure 6.4: The RF switch setup with labelled ports

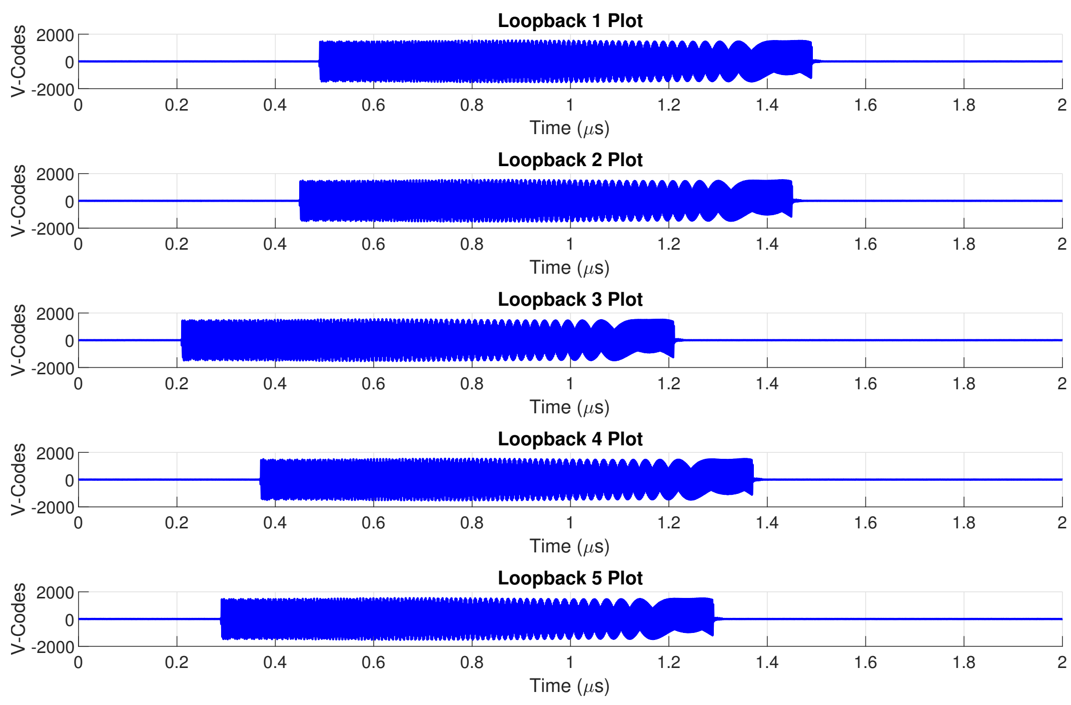


Figure 6.5: The loopback sequences $d_i[n]$ for 5 different system resets

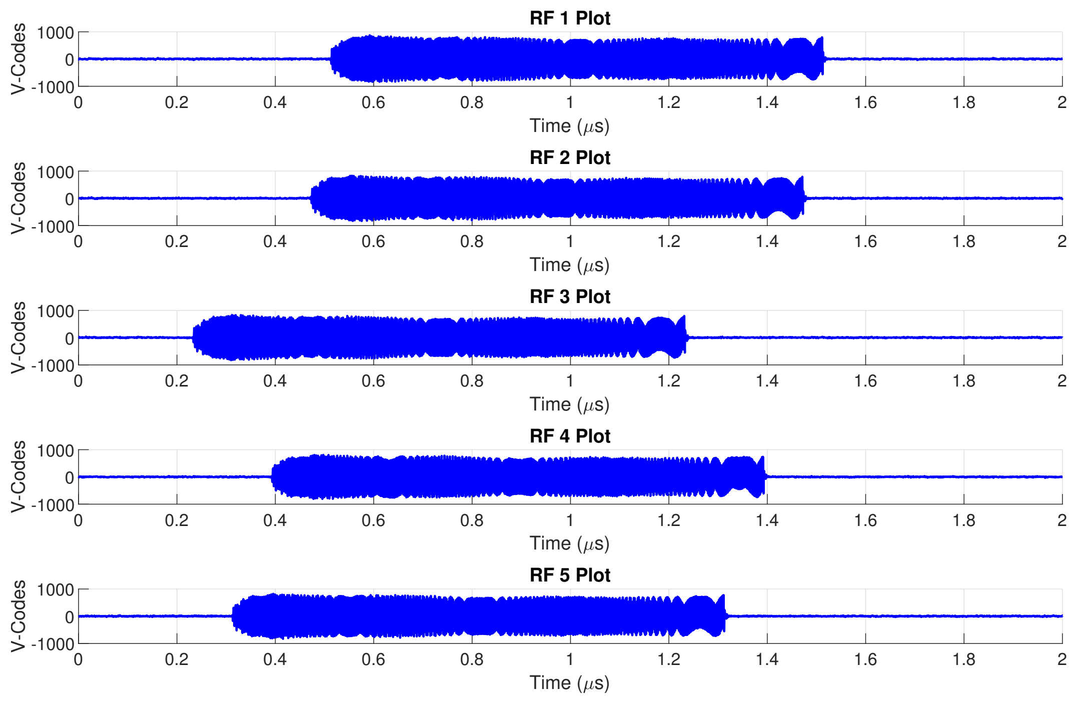


Figure 6.6: The RF calibration sequences $r_i[n]$ for 5 different system resets

If these datasets were to be pulse compressed without any compensation, they would produce peaks that vary in time (or range) by a large amount. Using the loop-back calibration technique allows the propagation delay through the system to be eliminated and for the change in τ_{dig} to be compensated for. A comparison between the "default" uncalibrated pulse compression of each RF dataset and the calibrated pulse compression of each RF dataset is shown in Figure 6.7. The uncalibrated pulse compressed peaks occur with $\sim$40 m of range variation. After calibration, the peaks line up nearly perfectly with one another with around 1 cm of deviation between them, which corresponds to the pulse-to-pulse latency variation of 60 ps. Notice also that the peaks appear at 0.65 m in range (denoted in Figure 6.7 as a black dashed line), which corresponds to the 1.3 m cable range because it is calculated

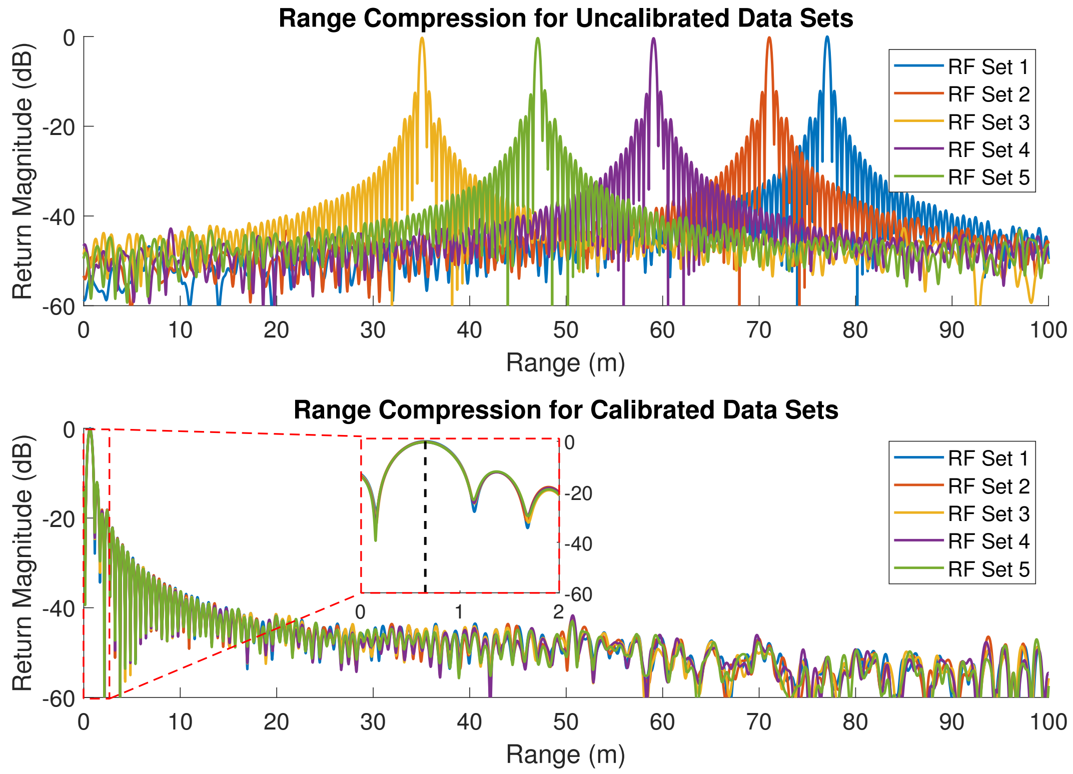


Figure 6.7: The uncalibrated pulse compression of the 5 datasets (top) vs. the calibrated pulse compression of the datasets (bottom)

assuming 2-way propagation as in (2.1).

6.5 Summary

In this chapter, a loopback calibration method was derived and implemented to account for variation in digital back-end latency between separate system resets. The loopback calibration methodology is proven theoretically to give the necessary calibration information, and the practical hardware setup is designed. When implemented, it is shown that the calibration method can sufficiently account for variation in latency between different start-ups. With this fix in place, the radar is coherent from pulse-to-pulse and from reset-to-reset, meaning that it can consistently measure ranges to targets accurately.

Chapter 7

Final Improvements and Sensing Results

With the radar operating as expected with the correct waveform properties, PRF, and strict timing between the DDS and ADC/FPGA, it is suitable for use. An image of the completed radar hardware is shown in Figure 7.1. Additionally, a simplified block diagram of the completed radar system is shown in Figure 7.2. This block diagram omits some details of the IF chain for simplicity, but indicates how the parts of the radar will be broken up in future miniaturizations of the system hardware. To make the radar more usable, several additional items are implemented and are briefly described in this chapter. These additional changes do not fundamentally alter the operation of the radar in any way, but they allow the radar to be useful both in a pulse-Doppler and a SAR application. This chapter also describes a test performed with the completed radar system to verify the pulse-Doppler operation of the system.

7.1 FPGA Firmware for Pulsed Captures

The most important feature added to the radar operation is an augmentation to the FPGA firmware allowing for pulsed captures of data. By default, the FPGA will capture one set of data at a time with an arbitrary number of samples captured.

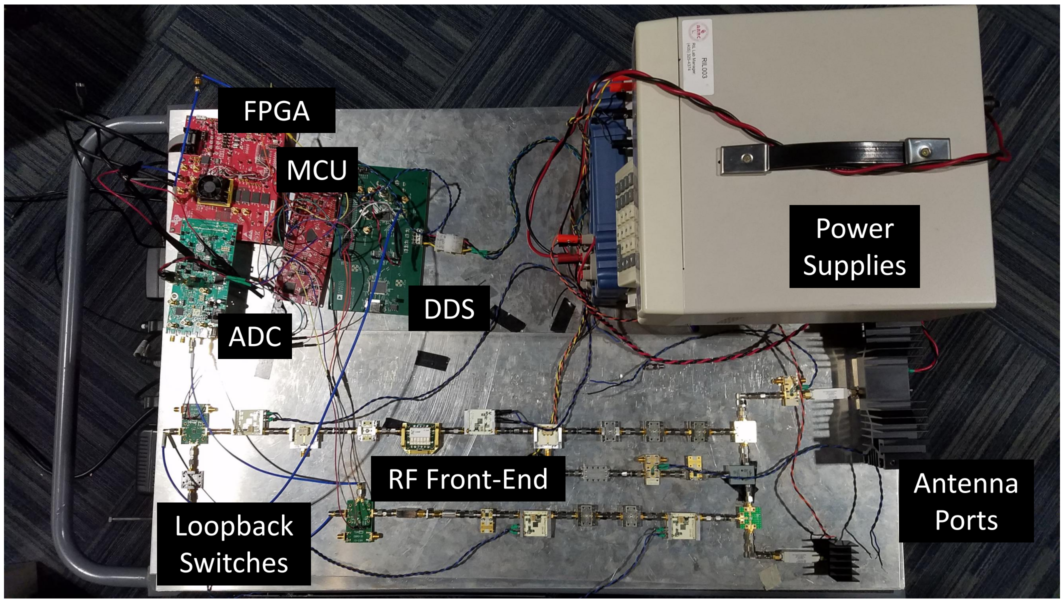


Figure 7.1: The completed radar hardware

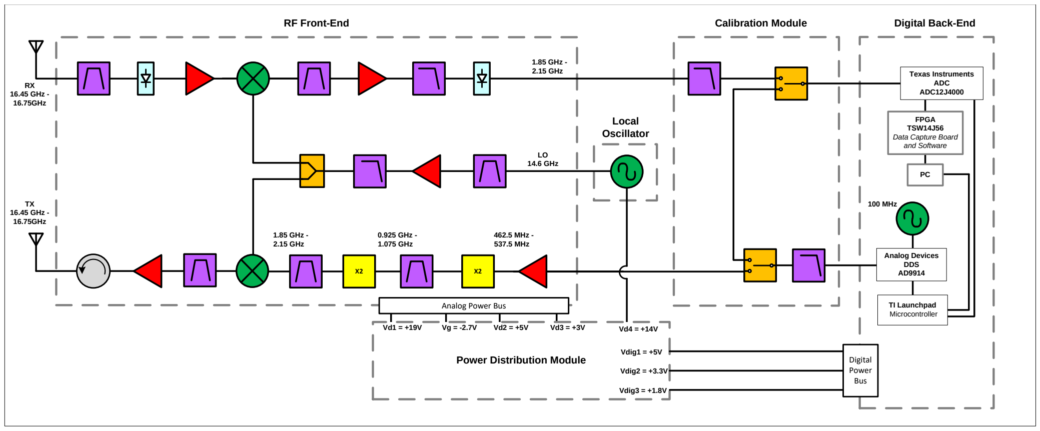


Figure 7.2: A simplified block diagram of the completed radar system.

Even by using the maximum amount of RAM on the FPGA (4 GB), the radar will only be able to capture 2 billion samples, or 1.25 s of data, which is not enough to form a significant synthetic aperture even at a high speed. Furthermore, most of the captured data is not useful. Given an expected flight altitude of 1.5 km and a look-angle of 30°, the radar returns are expected to be significant from 10-13 μs after being transmitted. This implies that the remaining 320 μs of data per pulse are not useful.

Therefore, it is desireable to modify the FPGA firmware to capture 2.5-20 μs of data each time it is triggered by the PRF signal and append each “mini-capture” into a single dataset to be parsed into individual pulses during processing. This means that longer timeframes of data (5-10 s) can be captured using only a fraction of the RAM on the FPGA. This feature was implemented in the FPGA firmware, allowing between 4080 and 16320 samples (2.55-10.2 μs) per mini-capture and between 1000 and 30208 mini-captures (0.33-10.07 s) in total. Furthermore, recall that the PRF signal used to initiate *IOUPDATE* is not restricted to be 1 μs in length. Therefore, by configuring the FPGA to be triggered on the falling edge of this PRF signal rather than the rising edge (see Section 4.3.3) and by increasing the length of the *IOUPDATE* PRF signal, the data capture can be configured to begin at an arbitrary time. A pulse train of 10 out of 1000 captured pulses with 4080 samples captured during each mini-capture is shown in Figure 7.3. The pulses are captured through the loopback switches, and as such the the waveform is at 500 MHz and

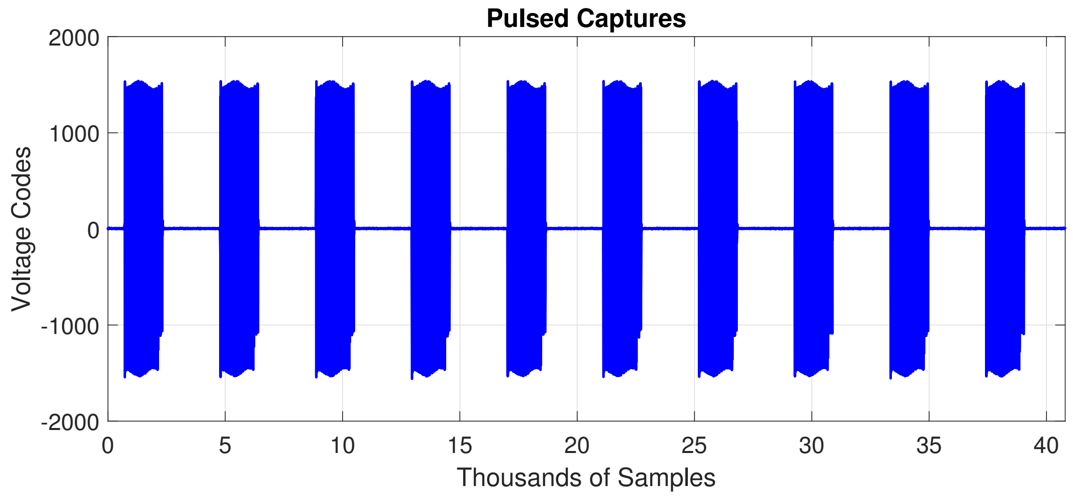


Figure 7.3: The first 10 pulses of a pulse train captured with 4080 samples per capture

the measured time is expected to be constant for each pulse.

To show that the FPGA firmware modifications do not introduce incoherence from pulse to pulse, a range-pulse map centered at 0 m is shown in Figure 7.4, indicating that the measured time (or range) does not vary from pulse to pulse. Additionally, the range-Doppler map of the CPI in Figure 7.5 shows that the pulses integrate coherently to give a peak at 0 m in range and 0 m/s in velocity.

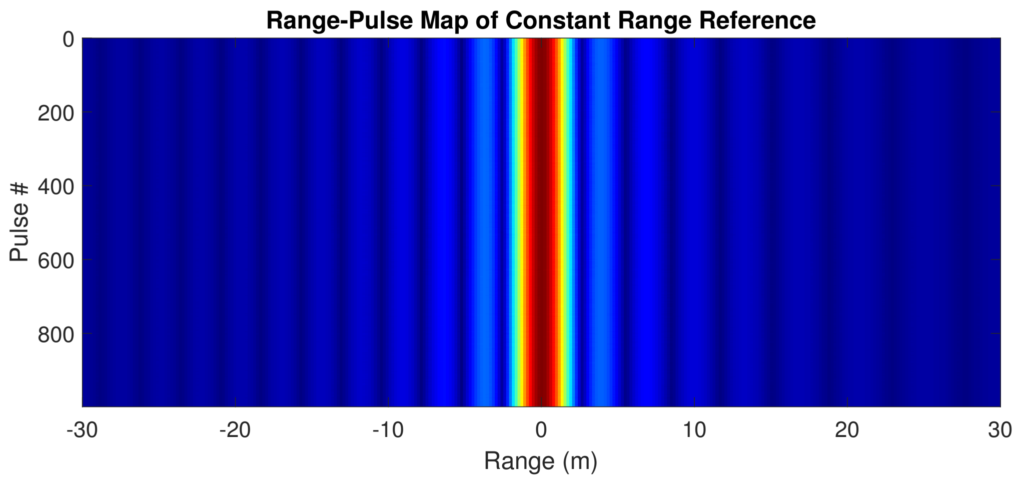


Figure 7.4: A range-pulse map of the pulsed capture of 1000 pulses centered at 0 m showing that the measured range remains constant throughout the CPI

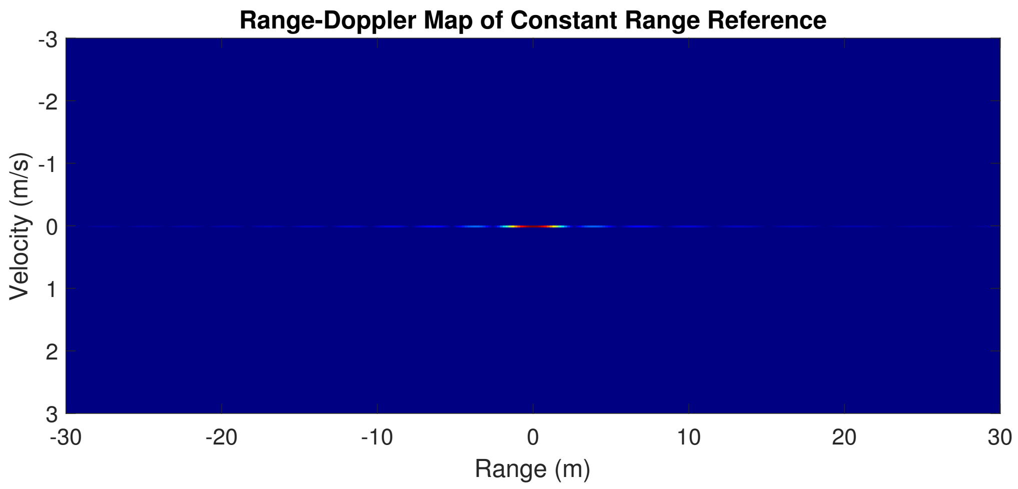


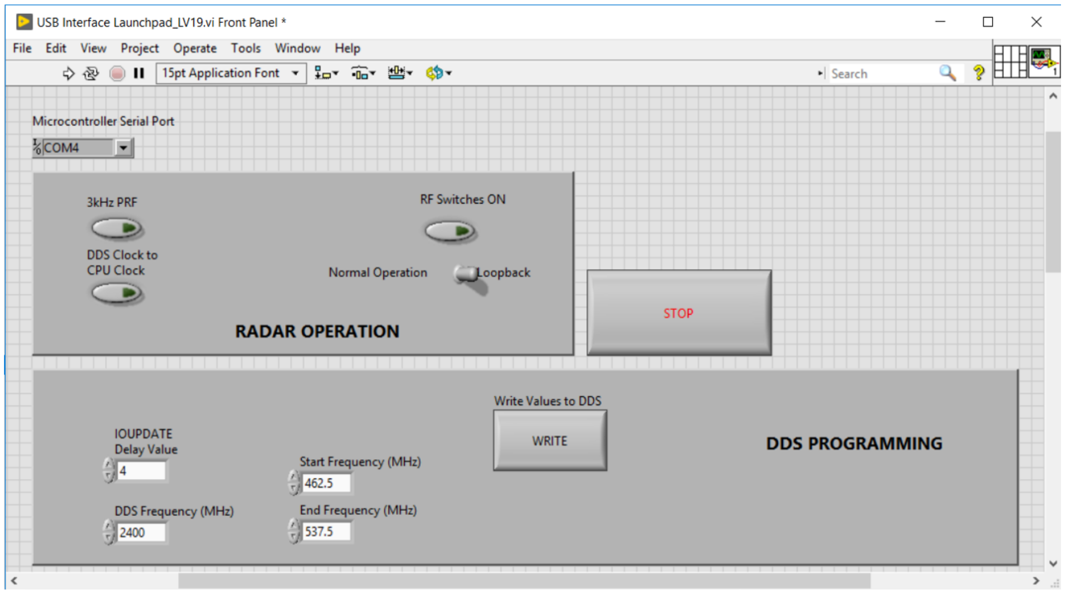
Figure 7.5: A range-Doppler map of the pulsed capture of 1000 pulses centered at 0 m showing that all 1000 pulses integrate coherently

7.2 LabVIEW GUI

Because the microcontroller USB port can be interfaced with as an RS-232 COM port, it can be communicated with via NI VISA through a LabVIEW application. To make the operation of the radar feasible from a high level, a LabVIEW GUI is developed to instruct the radar to perform certain core functions. An image of the GUI is shown in Figure 7.6.

The GUI contains controls for operating the radar during runtime and for configuring the DDS. The controls for radar operation are as follows:

- **Microcontroller Serial Port** - Allows the user to select the correct COM port that the microcontroller is connected to, since this may vary between different computers. Defaults to "COM 4"
- **3 kHz PRF** - Enables or disables the PRF triggering output by the microcontroller. Defaults to off



Microcontroller Serial Port: COM4

IOUPDATE Delay Value: 4

DDS Frequency (MHz): 2400

Start Frequency (MHz): 462.5

End Frequency (MHz): 537.5

Figure 7.6: An image of the LabVIEW GUI for controlling the radar

- **DDS Clock to CPU Clock** - When enabled, causes the microcontroller CPU to be clocked by the external 100 MHz f_{timer} clock provided by the DDS. Should be enabled before calibrating the ADC but disabled if the DDS is restarted. Defaults to off

- **RF Switches ON** - When disabled, the switches are configured to pass all power input to the switch directly to a 50 Ω termination. When enabled, power flow is controlled by the Normal Operation/Loopback control. Defaults to off

- **Normal Operation/Loopback** - Controls whether the loopback switches allow power through the loopback cable or to pass to the RF front-end for normal operation. Defaults to “Loopback”

The controls for programming the DDS are as follows:

- **Write Values to DDS** - When this button is pressed, the DDS registers are sent to the microcontroller, which programs the DDS through SPI

- **Start Frequency (MHz)** - Allows the user to modify f_{LO} in the chirp in MHz. Defaults to 462.5

- **End Frequency (MHz)** - Allows the user to modify f_{HI} in the chirp in MHz. Defaults to 537.5

- **IOUPDATE Delay Value** - Specifies the number of clock cycles of the microcontroller CPU clock by which the *IOPUDATE* PRF trigger should be offset from the *OSK* PRF trigger. Defaults to 4

- **DDS Frequency (MHz)** - Specifies the frequency of the DDS in MHz. Defaults to 2400

In addition to the basic controls, the user can modify the raw register values before programming the DDS, allowing for enabling or disabling matched latency, enabling or disabling the internal DDS PLL, and any other features of the DDS not discussed in this thesis. Note that for the radar operation described in this thesis, the default configuration options for the DDS should not be modified.

When configuring the radar, the typical startup sequence is as follows:

1. Power on all components of the radar
2. Program the DDS
3. Enable "DDS Clock to CPU Clock" to change the microcontroller clock
4. Use the ADC GUI to program the ADC
5. Use HSDC Pro to load firmware onto the FPGA
6. Enable "RF Switches ON" while leaving the switches configured to "Loopback"
7. Enable "3 kHz PRF" to begin pulsing. Capture a dataset in HSDC Pro and save it to be used as $d_1[n]$ in calibration during processing
8. Disable "3 kHz PRF." Flush the FPGA RAM so that the RAM will begin filling with new pulses once "3 kHz PRF" is enabled again
9. Switch from "Loopback" to "Normal Operation"
10. Enable "3 kHz PRF." The pulse data saved by the FPGA will begin as soon as the PRF is enabled
11. Load the data from the FPGA to HSDC Pro and save it for processing

After performing range calibration as described in Section 6.2.2, the saved data can be processed into range-Doppler maps and eventually SAR images. Examples of measured range-Doppler maps are given in Section 7.4.

7.3 IMU Triggering

To perform SAR imaging, the precise location of the radar with respect to the scene of interest must be known during each pulse. While it is theoretically possible to predict the motion path of the radar through estimation from kinematics, airplanes and other aerial platforms cannot realistically follow an ideal path due to unpredictable disturbances. To accurately monitor the location of the radar during flight, a device called an inertial measurement unit (IMU) is used. The IMU measures acceleration and angular velocity, and can interpolate position with varying degrees of accuracy by fusing these measurements with GPS data through an algorithm such as the Kalman Filter, as in [2] and [33]. The precision of the IMU should be comparable to the wavelength at the carrier frequency, since the motion compensation must account for small changes in received signal phase due to the change in position on each pulse [34].

Given the 1.8 cm wavelength for the operation of this system, an IMU with cm level accuracy is desirable. This is accomplished using the NovAtel Synchronous Position, Attitude and Navigation (SPAN) IMU-ISA-100C™. When fused with GPS through the ProPak6™ receiver and post-processed, the typical error in the IMU measurements is expected not to exceed 1 cm [35]. The IMU, the ProPak6, and the GPS antenna used in the radar are shown in Figure 7.7.

Because the IMU can only provide information at a maximum of 200 Hz, it is not practical to trigger it on every pulse to capture the position of the radar [35].

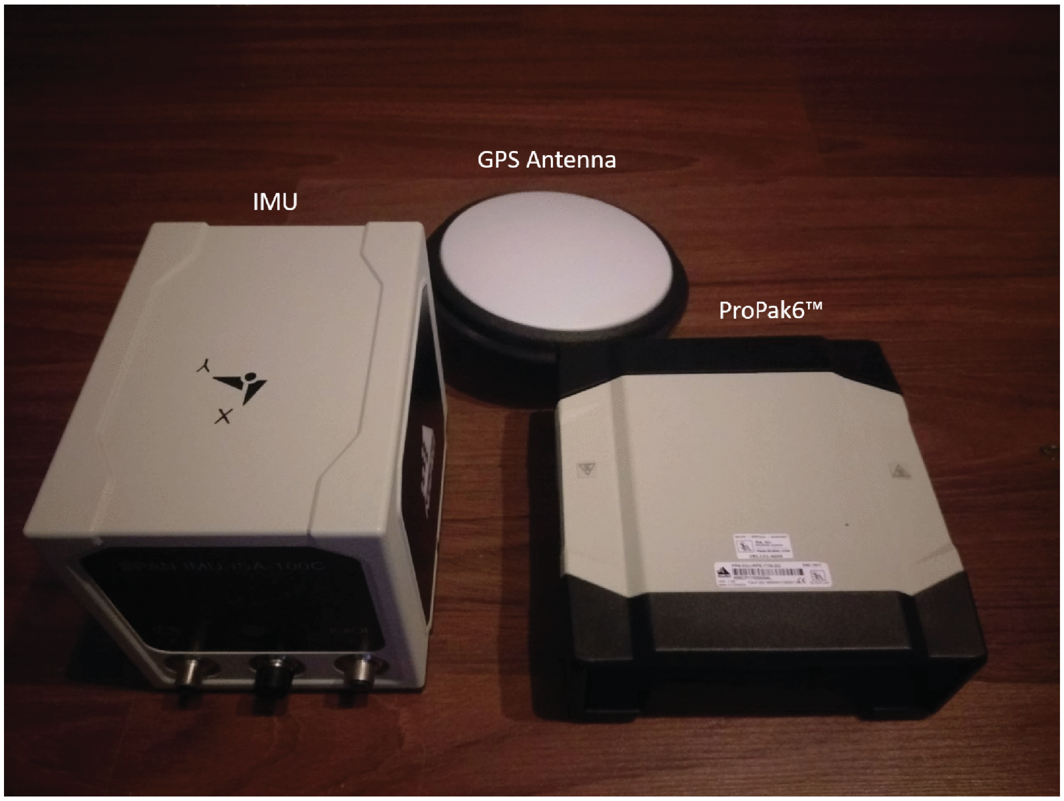


Figure 7.7: An image of the IMU, the GPS and data receiver, and the GPS antenna taken from [2]

Instead, the IMU is triggered to begin capturing position data on the first pulse of the radar. With the PRF being known and the time of the first pulse being correlated with the first measurement of the IMU, the data can be upsampled through some method of interpolation to obtain the correct position data at the time of each pulse.

The IMU and ProPak6 have several event inputs and outputs, allowing the devices to interact with other hardware through timed event pulses. The trigger inputs are interfaced through an I/O port DB9 connector on the ProPak6. The ProPak6 can be configured to log the inertial state of the IMU and the GPS time when triggered by one of the inputs (called "marks") [36]. To trigger the ProPak6, Mark1 is used, which is associated with pin 4 of the I/O port [37]. The ProPak6 is configured in the firmware to log the position and GPS time when an event is received on Mark1.

Because the long DB9 cable potentially adds capacitance that may affect the quick rise times of the *IOUPDATE* and *OSK* triggers, a separate PWM signal that is identical in configuration and timing to the *OSK* trigger is used on the microcontroller to generate the trigger. The DB9 cable is cut and broken out on one end, and the conductor attached to pin 4 is connected to this PWM channel. This enables the position measurements of the IMU to be logged and associated with the pulse times of the radar, allowing the data to be motion compensated to form SAR images.

7.4 Measured Field Results

To test the operation of the radar in a realistic environment, an outdoor field test is performed. The radar is used in a pulse-Doppler mode to measure the range and velocity of two moving trucks in the main beam of the radar. The trucks are travelling in opposite directions along a radial range line in the main beam of the antenna. One truck is driving away from the radar at 15 mph measured by the speedometer (6.7 m/s), while the other truck is driving toward the radar at 20 mph measured by the speedometer (8.9 m/s). The setup of the radar for this test is shown in Figure 7.8.

The data is captured using a firmware set that captures 15360 mini-captures with 4080 samples per mini-capture, giving 2.55 μs of capture time per mini-capture and a total CPI time of 5.12 s. To employ the processing techniques discussed in Section 2.1.1, the real data captured by the ADC is first reduced to complex baseband data by use of the Hilbert transform and a digital mixing with the 400 MHz signal band center frequency. The complex data is then pulse compressed to give the range-pulse map shown in Figure 7.9. The change in range of the trucks as they move from pulse to pulse is evident in this image. After Doppler processing, the

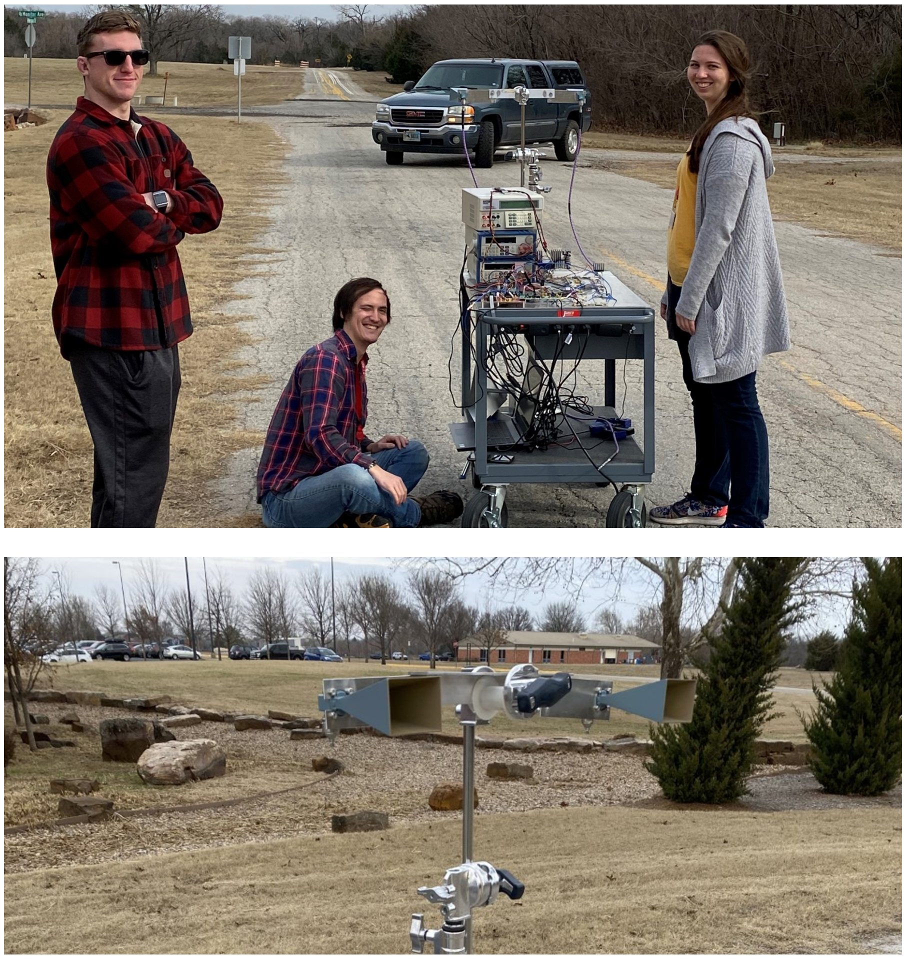


Figure 7.8: The radar set up outside (top) and the mounted quasi-monostatic horn antennae (bottom)

range-Doppler map shown in dB in Figure 7.10 is produced. Clearly, the trucks cannot be focused into a single range bin due to range migration over the course of the fairly long CPI. To account for the range migration, the Keystone Transformation described in [38] is used. This transformation modifies the data in the frequency domain to account for linear range migration, allowing targets smeared in range over the course of a CPI to be focused into a point. The point for a moving target after the Keystone Transformation will appear at the range the target is located at the middle of the CPI. The range-Doppler map of the scene after the Keystone Transformation is shown with a Google Earth image of the physical scene in Figure 7.11.

The results shown in Figure 7.11 indicate moving targets with velocities of 16.3 mph (7.3 m/s) away from the radar and 19.2 mph (8.6 m/s) toward the radar.

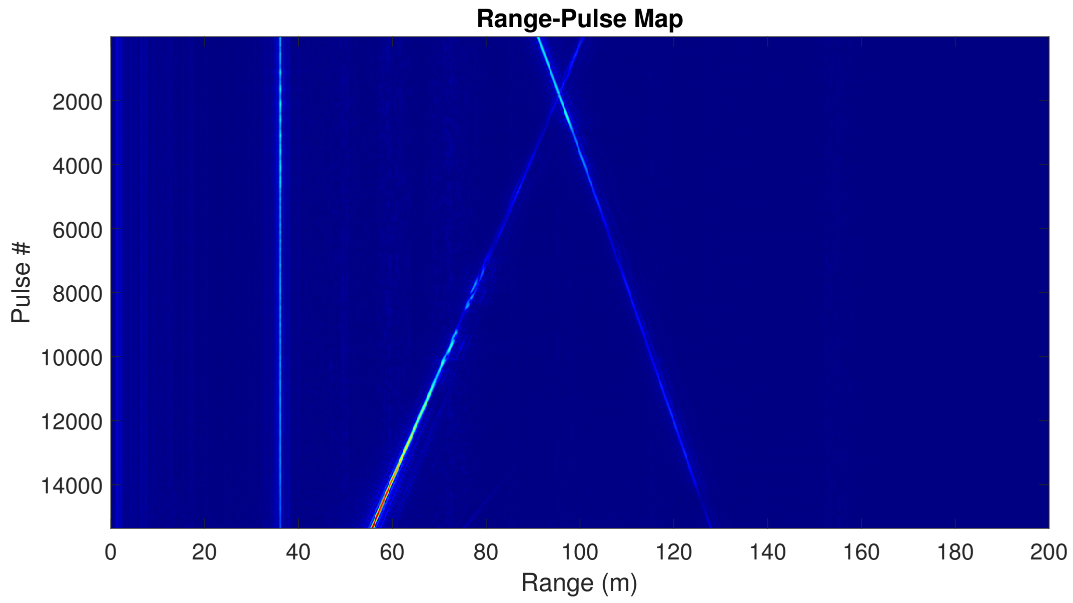


Figure 7.9: The range-pulse map of the 5.12 s CPI

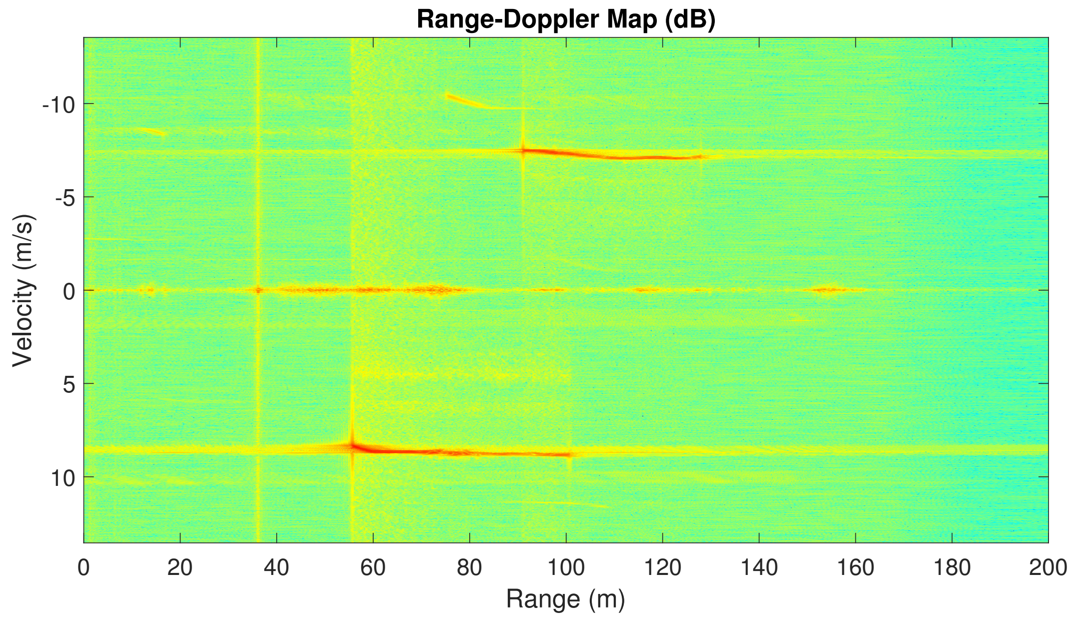
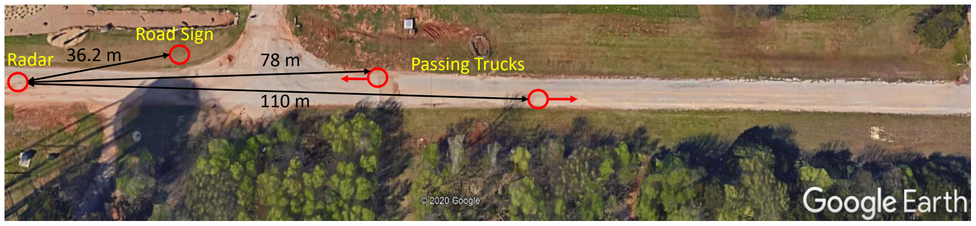


Figure 7.10: The range-Doppler map (in dB) of the 5.12 s CPI



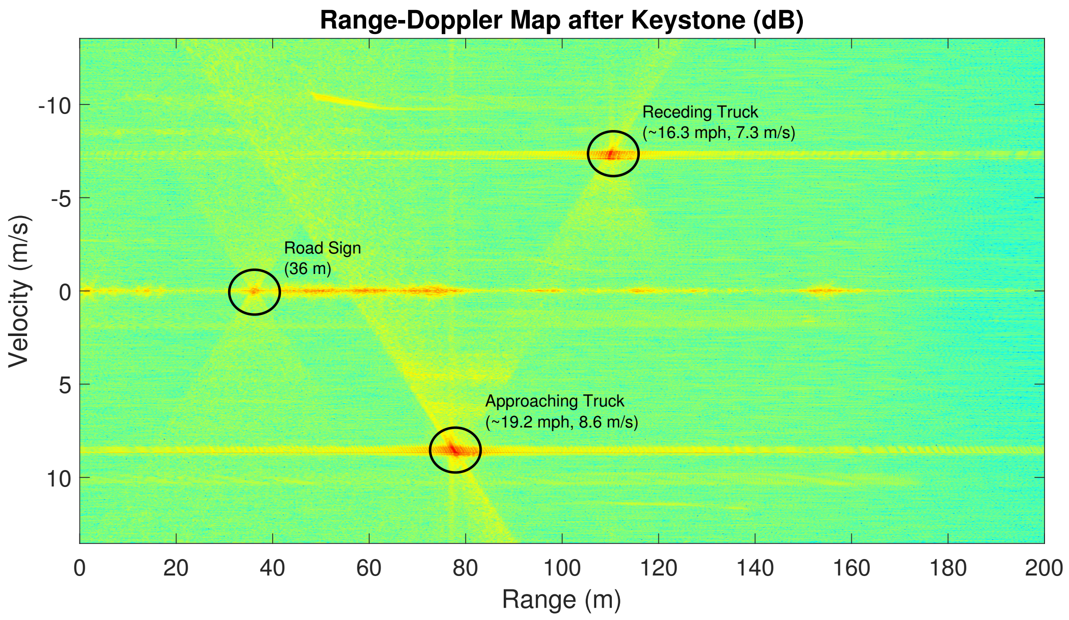


Figure 7.11: A Google Earth image of the target scene (top) and the range-Doppler map of the scene with the Keystone Transformation applied to the data (bottom)

These values are not exactly the values measured using the speedometers of the trucks but taking into account measurement and driver error they are close enough to verify the radar's operation. In the clutter caused by the ground, a road sign is also captured by the radar with a strong return at 36 m. Measuring from the radar location to the sign verifies that this is correct to within a range bin.

7.5 Summary

In this chapter, final improvements and operational modifications to the radar were presented. These modifications, while perhaps somewhat trivial in implementation, are crucial to the operation of the radar both in the basic pulse-Doppler mode of operation and also in the eventual SAR application. Furthermore, measured results from the radar taken in a noisy environment with real targets were demonstrated. These results indicate the radar's ability to resolve targets both in range and in Doppler. The feasibility of Doppler processing confirms the PRF is set properly, and the lack of range drift from pulse to pulse beyond the range migration indicates that the radar is suitably coherent from pulse to pulse. More broadly, these results indicate that in the future, the radar is usable for SAR data collection and processing.

Chapter 8

Conclusion and Future Work

At the Advanced Radar Research Center, a pulse-Doppler radar system has been designed and constructed using exclusively COTS parts for application as a low SWaP-C SAR system for imaging applications onboard lightweight aerial vehicles. The contribution of this thesis is the implentation of a coherent digital transceiver using high-frequency digital evaluation modules. A control system is designed for the radar to regulate the generation of ideal LFM pulses that can be transmitted at a desired PRF without spectral distortion or timing jitter from pulse to pulse. A clocking system and modified RF front-end are designed as well to ensure system coherence and phase-locked behavior between the DDS and ADC. The problem of startup timing calibration is dealt with using an original RF switch loopback scheme, whose operation is proven both theoretically and practically. In January of 2020, the radar was used in a field test to generate accurate range-Doppler maps of moving targets, indicating the radar's ability to resolve scatterers both in range and in Doppler shift.

Moving forward, there are multiple improvements that can be made to the system to make it more useful, as well as additional data that can be obtained. The first improvements involve enhancing the automation of the radar data capture. The ADC chip, the TRF3765, and the LMK04828 each can be programmed without the

ADC GUI through a serial interface similar to the DDS. Programming the micro-controller to program these chips on startup with the DDS would eliminate usage of the ADC GUI and simplify the startup process. Additionally, a DLL exists that allows commands to be issued to HSDC Pro automatically to load firmware, configure the FPGA, and capture and save data. This DLL could be incorporated into LabVIEW, which would allow the entire operation of the radar to be performed from a single user interface.

One benefit of the digital radar system as it is designed is that it can be reconfigured to arbitrary specifications of operation given an appropriately modified RF front-end. The LFM bandwidth at the output of the DDS can be arbitrarily increased to 960 MHz, while the maximum bandwidth that can theoretically be absorbed by the ADC is 2 GHz, enabling very high bandwidth and fine range-resolution. Additionally, the PRF and pulse width can also be set arbitrarily. A future improvement would be to add controls to the LabVIEW GUI to allow for reconfiguration of these characteristics.

To enable the placement of the radar onto a flight platform, the back-end evaluation boards must be organized into a small form-factor chassis of some sort. Additionally, there are currently plans for fabricating a PCB to fully integrate the RF front-end and include a custom power supply. This will significantly reduce the size and weight of the radar and make it feasible to fly on a plane to generate airborne SAR images.

To further prove the pulse to pulse coherence of the radar and the loopback calibration, it is desired to obtain range measurements through an optical delay line. An optical delay line takes an RF input and upconverts the signal to optical frequencies and then transmits the optical signal through a fiber optical cable with a precisely known length. At the end of the optical cable, the signal is downconverted again to the original RF frequency and output from the delay line chassis. This allows the radar transceiver to be tested with a precisely known delay time, enabling exact calibration of the radar's range measurements.

Finally, as mentioned in Section 7.5, the radar is currently capable of performing SAR imaging given its proper pulse-Doppler operation and the ability to integrate with the IMU for motion compensation. It is currently planned to use the radar in a SAR ground test to form an image of the Gaylord Family Oklahoma Memorial Stadium. The image can be formed using a fairly short synthetic aperture given that the distance from the antenna to the scene center of the stadium image will be quite small.

References

[1] *ADC12J4000 12-Bit, 4-GSPS ADC with Integrated DDC*, Texas Instruments, Oct. 2017, Rev. D.

[2] B. Sun, M. Yeary, H. H. Sigmarsson, and J. W. McDaniel, "Fine Resolution Position Estimation Using Kalman Filtering," in *2019 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, 2019, pp. 1–5.

[3] M. I. Skolnik, *Introduction to Radar Systems*, 2nd ed. McGraw-Hill, 1980.

[4] J. W. McDaniel, "Design, Integration, and Miniaturization of a Multichannel Ultra-Wideband Snow Radar Receiver and Passive Microwave Components," Master's thesis, University of Kansas, 2015.

[5] M. A. Richards, *Fundamentals of Radar Signal Processing*, 2nd ed. McGraw-Hill Education, 2014.

[6] Space Dynamics Laboratory, "NuSAR: Naval Research Laboratory (NRL) Unmanned Aerial Vehicle (UAV) Synthetic Aperture Radar."

[7] K. W. Sorensen and R. Riley, "Low SWaP Radars for Manned and Unmanned Systems," Sandia National Laboratories, Feb. 2017.

[8] E. Murphy and C. Slattery, "All About direct Digital Synthesis."

[9] JEDEC, *Serial Interface for Data Converters*, December 2017.

[10] S. Murthy, "Implementing JESD204B SYSREF and Achieving Deterministic Latency With ADC32RF45," Texas Instruments, Inc., 2016.

[11] J. McDaniel, "Ku-Band Synthetic Aperture Radar: System Parameters Study," 2017.

[12] *3.5 GSPS Direct Digital Synthesizer with 12-Bit DAC*, Analog Devices, Feb. 2017, Rev. F.

[13] *TSW14J56 JESD204B High-Speed Data Capture and Pattern Generator Card User's Guide*, Texas Instruments, Jan. 2016, Rev. C.

[14] J. Brinkhurst, *ADC12J4000EVM Schematic*, Texas Instruments, Sept. 2014.

[15] *TRF3765 Integer-N/Fractional-N PLL With Integrated VCO*, Texas Instruments, Dec. 2015, Rev. E.

[16] *LMK0482x Ultra Low-Noise JESD204B Compliant Clock Jitter Cleaner with Dual Loop PLLs*, Texas Instruments, Dec. 2015, Rev. AR.

[17] *ADF4351 Wideband Synthesizer with Integrated VCO*, Analog Devices, Jan. 2017, Rev. A.

[18] *M Series User Manual: NI 622x, NI 625x, and NI 628x Multifunction I/O Modules and Devices*, National Instruments, July 2016.

[19] J. F. Kaiser, "On a simple algorithm to calculate the 'energy' of a signal," in *International Conference on Acoustics, Speech, and Signal Processing*, April 1990, pp. 381–384 vol.1.

[20] K. G. (2011, Apr.) AD9910 DROVER. Analog Devices. [Online]. Available: https://ez.analog.com/dds/f/q-a/30274/ad9910-drover

[21] *LPC176x/5x User Manual*, NXP Semiconductors, Dec. 2016, Rev. 4.1.

[22] K. G. (2013, Jan.) AD9914 Multichip Sync. Analog Devices. [Online]. Available: https://ez.analog.com/dds/f/q-a/29717/ad9914-multichip-sync

[23] *TMS320F28004x Piccolo™ Microcontrollers*, Texas Instruments, Oct. 2018, Rev. D.

[24] *C2000™ Piccolo™ F28004x Series LaunchPad™ Development Kit*, Texas Instruments, June 2018.

[25] *TMS320F28004x Piccolo Microcontrollers: Technical Reference Manual*, Texas Instruments, Jan. 2019.

[26] J. Brinkhurst. (2019, Feb.) TSW14J56EVM: ACD12J4000 - Changing JESD204B K-value and SYSREF Frequency. Texas Instruments. [Online]. Available: https://e2e.ti.com/support/data-converters/f/73/t/771653

[27] ——. (2019, Feb.) TSW14J56EVM: Utilizing SYSREF Input to FPGA to Synchronize with Other Device. Texas Instruments. [Online]. Available: https://e2e.ti.com/support/data-converters/f/73/t/768849

[28] *LFCN-575 Ceramic Low Pass Filter*, Mini-Circuits, Rev. L.

[29] *SYK-2R+ 2X Frequency Multiplier*, Mini-Circuits, Rev. D.

[30] *GVA-62+ Flat Gain, High IP3 Monolithic Amplifier*, Mini-Circuits, Rev. A.

[31] *CBP-1000+ Surface Mount Bandpass Filter*, Mini-Circuits, Rev. OR.

[32] *HSWA2-63DR+ MMIC SP2T RF Switch*, Mini-Circuits, Rev. OR.

[33] Honghui Qi and J. B. Moore, “Direct Kalman filtering approach for GPS/INS integration,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 2, pp. 687–693, 2002.

[34] G. W. Stimson, *Introduction to Airborne Radar*, 2nd ed. SciTech Publishing, Inc., 1998.

[35] *SPAN IMU-ISA-100C*, NovAtel, May 2016, v5.

[36] *SPAN® on OEM6® Firmware Reference Manual*, NovAtel, Dec. 2016, Rev. 8.

[37] *SPAN® on OEM6® User Manual*, NovAtel, Feb. 2017, Rev. 12.

[38] M. Richards, “The Keystone Transformation for Correcting Range Migration in Range-Doppler Processing,” 2014.