X-Band Doppler Simulator for Sport Projectile Radars

Binjamin Barsch

A dissertation submitted to the Department of Electrical Engineering, University of Cape Town, in partial fulfilment of the requirements for the degree of Master of Engineering.

Cape Town, September 2017

Declaration

I declare that this dissertation is my own, unaided work. It is being submitted for the degree of Master of Science in Engineering in the University of Cape Town. It has not been submitted before for any degree or examination in any other university.

I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own. This dissertation has been submitted to the Turnitin module and I confirm that my supervisor has seen my report and any concerns revealed by such have been resolved with my supervisor.

Signature of Author 

Cape Town
1 September 2017

Abstract

Systems engineering has always required that hardware is evaluated in its desired environment. However, this may not be feasible as the target or environment may be too complex or too costly to use at any given time. This is a common problem with evaluating Doppler radars as well, since the inherent property of a Doppler radar is to measure the radial velocity of objects in motion like aircraft or projectiles. A common solution to this problem is to perform a hardware-in-the-loop (HIL) simulation. This usually comprises of a device that does a real-time simulation of the environment or moving target. In the field of RF engineering, such a device is known as a repeater or a Doppler simulator. Depending on the application, these devices use either the digital radio frequency memory (DRFM) or direct digital synthesis (DDS) simulation method. Developing Doppler simulators as a diagnostic tool for sport Doppler radars is a growing need to evaluate and assess the performance of these radars.

This dissertation will investigate the design and development of a Doppler simulator that can be used to simulate projectiles for sport Doppler radars. The scope of this dissertation was restricted to the sport of golf using continuous wave (CW) X-band Doppler radars. Raw data was measured by a Doppler radar to determine the velocity profiles of golf balls in flight. From these profiles, flight models were developed that could be simulated using a Doppler simulator. An Arduino Due microcontroller was used to implement the digital DDS method and to simulate these velocity profiles. This microcontroller was integrated into an existing Doppler simulator that lacked the capabilities to simulate a velocity profile.

Results showed that the projectile based sport Doppler simulator was effective in simulating the modeled flight trajectories. A close comparison between the simulated and measured result were shown. For three different types of golf shots, the average error between the simulated and measured trajectories was -0.169 m/s while the standard deviation was 0.28 m/s. This dissertation also showed future possibilities in simulating a diverse range of projectiles and targets.

Acknowledgements

I would firstly like to thank my parents for their continuous support and encouragement. Then to my supervisor, Prof. Daniel O'Hagan for his guidance throughout my research. I would also like to thank my mentors and colleagues at FlightScope for their assistance, support, and knowledge in the art of science and engineering. Finally, I need to thank Bob Rust for proposing the dissertation topic and for his patience and guidance in answering my endless questions.

Contents

Declaration i

Abstract ii

Acknowledgements iv

Contents v

List of Figures ix

List of Tables xiv

List of Symbols xvi

Nomenclature xvii

1 Introduction 1

1.1 Design Requirements 3
1.2 Functional Specifications 3
1.3 Design and Development Methodology 4
1.4 Scope and Limitations 5
1.5 Executive Summary 5

2 Literature Review 7

2.1 The Fundamentals of Doppler Radars 7
2.2 RF Frequency Synthesis 10
2.2.1 RF Mixer 10

2.2.2 In-Phase and Quadrature Signals 13
2.3 The Principles of DDS 14
2.3.1 Look-Up Table 15
2.3.2 Phase Accumulator 16
2.3.3 Phase Incrementor 17
2.3.4 Digital-To-Analog Converter 18
2.4 Golf Ball Flight Dynamics 18
2.4.1 Golf Terminology 18
2.4.2 Flight Dynamics 20

3 Feasibility Study **21**
3.1 Doppler Simulator Solutions 21
3.2 Processor Comparison 23
3.3 Chapter Summary 24

4 Design of the Due Doppler Simulator **25**
4.1 System Design 25
4.2 Analysis of the AD9 Doppler Simulator 26
4.3 Synthesis of the Due Doppler Simulator 30
4.3.1 System Operation 30
4.4 Flight Model Design 31
4.4.1 Empirical Simulation 31
4.4.2 Velocity Profile 32
4.4.3 Amplitude Profile 40
4.5 Firmware Design 42
4.5.1 Firmware Operation 43
4.5.2 Serial Communication 44
4.5.3 DDS Method Design 44
4.5.4 Waveform Modulation 48
4.6 Electronic Hardware Design 49
4.6.1 Arduino Due Overview 50
4.6.2 Passive Filter Design 51
4.6.3 Serial Communication 52

4.6.4 Power Consumption 53
4.7 Chapter Summary 54

5 Firmware and Hardware Integration 56
5.1 Firmware Implementation 56
5.2 Electronic Hardware Integration 61
5.3 Chapter Sumary 63

6 Evaluation of the Due Doppler Simulator 64
6.1 Evaluation under Controlled Environment 64
6.2 Evaluation with a Doppler Radar 67
6.2.1 Summary of the Results 74
6.3 Chapter Summary 75

7 Results And Findings 76
7.1 Due Doppler Simulator Integration 76
7.2 Simulated Velocity Profile Analysis 77
7.3 Device Feasibility 78
7.4 Chapter Summary 79

8 Conclusions And Recommendations 80
8.1 Future Considerations 80
8.1.1 Electronic Hardware Improvements 81
8.1.2 Firmware Improvements 81
8.1.3 Signal Processing Optimization 82
8.1.4 RF Design Updates 83
8.2 Conclusion 83
8.2.1 Contributions 83
8.2.2 Future Research 84

Bibliography 85

A Datasheets 91

B Doppler Simulator Specifications 93

C Doppler Simulator Testing **95**

D Bill of Materials **97**

E RF Test Equipment **98**

List of Figures

2.1 A representation of how the Doppler frequency changes whether it is moving towards or away from the device measuring it [21]. . . 8

2.2 The relationship between the range and signal power received by a target moving away from the radar [22]. 9

2.3 A block diagram of an RF upconversion mixer [24]. 12

2.4 A representation of the output of the RF mixer showing the lower and upper sidebands R_{F1} and R_{F2} around the carrier frequency LO. Through the SSB RF mixer the USB, R_{F2}, is suppressed as shown on right. 13

2.5 A representation of the I and Q waveforms showing how I is leading Q. 14

2.6 A flow diagram of the DDS method showing the NCO and DAC operation [31]. 15

2.7 A representation of the phase accumulator as a phase wheel and how it generates a sine waveform [29]. 17

2.8 The shaft loft [37]. 19

2.9 This figure shows the representation of the velocity and height profile versus range of a golf ball in flight. 20

4.1 A top view of the AD9 Doppler simulator 26

4.2 A layout of how the current AD9 Doppler simulator is used relative to a Doppler radar. 27

4.3 An image of the "HMC521" RF mixer showing the input pins of LO and IF1 and IF2. As well as showing the RF output pin. . . . 28

4.4 A block diagram showing the main components of AD9 Doppler simulator. 29

4.5 A functional flow diagram of how the AD9 Doppler simulator is operated and showing its various modes. 29

4.6 A top view of the RF board. 30

4.7 Flow diagram of the Due Doppler simulator function. 31

4.8 Raw Doppler data of a 6 Iron golf shot. 32

4.9 A spectrogram plot of the raw Doppler data of a 6 Iron golf shot. 34

4.10 A polynomial fit on the velocity profile of the 6 Iron golf shot. . . 35

4.11 Raw Doppler data of a Driver golf shot. 36

4.12 A spectrogram plot of the raw Doppler data of a Driver golf shot. 37

4.13 A polynomial fit of the spectrogram plot of the raw Doppler data of a Driver golf shot. 38

4.14 The raw Doppler data of a Pitching Wedge golf shot. 38

4.15 The spectrogram of the raw Doppler data of a Pitching Wedge golf shot. 39

4.16 The polynomial fit of the spectrogram of the raw Doppler data of a Pitching Wedge golf shot. 40

4.17 The amplitude profile of the three golf shots. 41

4.18 The exponential fit on the amplitude profile of the three golf shots. 42

4.19 The flow diagram for the firmware for the Due Doppler simulator. 43

4.20 The flow diagram for DDS method [57]. 45

4.21 A simulation of the DDS method. 48

4.22 A top view perspective of the Arduino Due 51

4.23 Bluetooth HC-06 Module [58] 53

4.24 A schematic diagram for the electronic components. 54

4.25 Electronic Hardware Layout 55

5.1 A snapshot of the serial monitor used to communicate with the Due Doppler simulator via USB 60

5.2 A snapshot of the app used to communicate via Bluetooth with the Due Doppler simulator 60

5.3 A top view perspective of the Arduino Due integrated into the housing. 62

5.4 A top view of the accessory board. 62

6.1 A snapshot of the spectrum analyzer of the AD9 Doppler simulator off while the test X-band hand is transmitting. 65

6.2 The raw Doppler data captured by the Doppler radar of 3.5 kHz simulated by the Due Doppler simulator 66

6.3 USB suppression relative to the LSB. 67

6.4 The Doppler radar setup relative to the Due Doppler simulator. . 68

6.5 The raw Doppler data captured by the Doppler radar of 3.5 kHz simulated by the Due Doppler simulator 68

6.6 The spectrogram of the raw Doppler data of the simulated 3.5 kHz 69

6.7 6 Iron Raw Doppler 70

6.8 6 Iron Spectrogram 70

6.9 The plot of the simulated 6 Iron shot versus the modeled and real data 71

6.10 Driver Raw Doppler 71

6.11 Driver Spectrogram 72

6.12 The plot of the simulated Driver shot versus the modeled and real data 72

6.13 Pitching Wedge Raw Doppler 73

6.14 Pitching Wedge Spectrogram 73

6.15 The plot of the simulated pitching wedge versus the modeled and real data 74

A.1 RF Mixer Pin Description 91

A.2 SAM3X8E DAC Characteristics 91

A.3 SAM3X8E DC DAC characteristics part 1 92

A.4 SAM3X8E DC DAC characteristics part 2 92

B.1 An illustration for the pattern of the patch array antenna. 93

C.1 Test points on the RF board showing where the IQ signals were measured. 95

C.2 Using an oscilloscope this shows a frequency spectrum of the two IQ signals showing 3.5 kHz. The discontinuity in the lagging signal is caused by the phase shifter board to create a slight phase adjustment to suppress the USB. 96

C.3 A snapshot of the spectrum analyzer for the AD9 Doppler simulator while the X-band antenna is transmitting. 96

D.1 The bill of materials for the Due Doppler simulator. 97

E.1 The Tektronix TDS1002B Oscilloscope. 98

E.2 The Agilent E4407B Spectrum Analyzer. 99

E.3 The Power Supply 99

E.4 The frequency synthesizer and reference oscillator 100

E.5 The two X-band horn antennas 100

List of Tables

2.1 A table of metrics that can be expected from professional golfers [37]. 19

3.1 A comparison of the two DDS processors 23

4.1 Spectrogram Parameters 33

4.2 Modeled golf shot comparison versus real data in Hz 39

4.3 Amplitude Retardation Values 41

4.4 A parameter list for the DDS process 47

6.1 A comparison between the modeled and simulated plots relative to the real plot for the 6 Iron golf shot 71

6.2 A comparison between the modeled and simulated plots relative to the real plot for the Driver golf shot 73

6.3 A comparison between the modeled and simulated plots relative to the real plot for the Pitching Wedge golf shot 74

6.4 A table comparing the average error and standard deviation for all three shots 75

7.1 A table comparing the velocity average error and standard deviation for all three shots 77

B.1 A table of specifications that is used in the Doppler Simulator . . 94

List of Symbols

B	—	Transmitted RF bandwidth
B_t	—	Total Radar Bandwidth
B_d	—	Doppler bandwidth of a single scatterer before azimuth dechirp
c	—	Speed of light
f_{ad}	—	Radar A/D sampling frequency
f_c	—	Radar centre transmit frequency
f_d	—	Doppler frequency
f_s	—	DDS sampling frequency
f_t	—	Tuning frequency
f_{res}	—	Frequency resolution
T_w	—	Tuning word
V_r	—	Radial velocity
λ	—	Wavelength
c	—	Speed of light

Nomenclature

Azimuth—A polar angle measured as the variation from a centre point parallel to the horizon.

Elevation—A Polar angle measured as the variation from a centre point perpendicular to the horizon.

Beamwidth—The angular width of a slice through the main lobe of the radiation pattern of an antenna in the horizontal, vertical or other plane.

Doppler frequency—A shift in the radio frequency of the return from a target or other object as a result of the object's radial motion relative to the radar.

Vertical Launch Angle—The vertical angle with which a projectile is launched from.

Polarization—Polarisation of an electromagnetic wave is defined as the orientation of the electric field with respect to the surface of the earth.

Range—The radial distance from a radar to a target.

Phase Array Antenna—A group of arrays that are placed a certain distance away from each other.

3dB Beamwidth—The angular width of the main lobe whereby the power levels remain above half the main lobes maximum power.

ATC—Air Traffic Control.

CW—Continuous Wave.

ADC—Analog-to-digital converter.

DAC—Digital-to-analog converter.

DCM—Down-conversion method.

UCM—Up-conversion method.

LO—Local oscillator.

IF—In-phase frequency.

IQ—In-phase and quadrature.

ISR—Interrupt Service Routine.

LUT—Look-up table.

PI—Phase Incrementor.

PA—Phase Accumulator.

DDS—Direct Digital Synthesis.

DRFM—Digital Radio Frequency Method.

DCM—Down-conversion method.

NCO—Numerically Controlled Oscillator.

DUT—Device under test.

DAC—Digital To Analog Converter.

SSB—Single-sideband mixer.

RPM—Revolutions per minute.

Chapter 1

Introduction

The development of engineering devices has always required the need for diagnostic tools to test and evaluate them. Similarly, in the field of RF engineering, radars need to be evaluated to assess their detection and tracking performance. However, this is sometimes difficult as the target environment or object may be too unpredictable or dynamic, like the weather, aircraft or projectiles. As in other industries, a solution to resolving this difficulty has been to develop a device that could perform a hardware-in-the-loop (HIL) simulation [1].

A HIL simulation comprises of a device that performs an end-to-end real-time simulation of the environment or target object in order to assess how the device would respond. In RF engineering, such a device is commonly known as a repeater or Doppler simulator [2][3]. These types of devices can use either the digital radio frequency memory (DRFM) or the direct digital synthesis (DDS) method [4][5]. These devices have also been used in electronic counter-measures as jammers or false target simulators by using the DRFM method [6]. These devices are also available commercially [7], and literature has been written about them extensively [8][9]. However, the focus of these devices has been dominated by pulsed radars for air-traffic-control (ATC) radars and military applications. Some research has gone into continuous wave (CW) Doppler radars for sport applications [10][11]. However, little research has gone into Doppler simulators needed to test and evaluate them. Furthermore, commercial Doppler simulators

[7] focus on fixed or linearly varying velocity profiles. However, sport Doppler radars, used in projectile sports, measure a varying velocity profile. Therefore Doppler simulators are required to simulate the velocity profile of projectiles in order to perform an effective HIL simulation for Doppler radars.

This dissertation investigates the design and development of a Doppler simulator for CW sport Doppler radars in order to perform an effective HIL simulation [12]. The research was performed at FlightScope [13]. FlightScope is a company that develop CW sport Doppler radars for a variety of sports, including golf. They have also developed a Doppler simulator for HIL simulation testing to evaluate their radars. However, their device, called the AD9 Doppler simulator, only simulates a constant velocity profile. Thus, this provides an opportunity to develop a Doppler simulator, focusing on the sport of golf, to simulate a varying velocity profile for golf ball projectiles. Furthermore, the flight dynamics of a golf ball has a generic projectile profile that can be used as a benchmark for other sport applications. These include soccer, tennis, and athletics.

The approach to develop a Doppler simulator used a variant of the DRFM method [14]. Measured golf ball flight data was attained from FlightScope Doppler radars and used to determine empirical flight models. In order to implement this approach, the firmware of the AD9 Doppler simulator needed to be either upgraded or its processor needed to be replaced. The latter approach was chosen as it allowed for further features to be implemented in the Doppler simulator. The Arduino Due microcontroller [15] was used to replace the existing processor. The upgraded device was called the Due Doppler simulator.

Results showed that the proposed approach was effective and the simulated trajectories versus the measured data had a close correlation. Three different types of golf shots were used to test the functionality of the device. These golf shots were referred to by their club as a Driver, a 6 Iron, and a Pitching Wedge. The average error between the simulated and measured trajectories was -0.169 m/s while the standard deviation was 0.28 m/s. This showed that the approach was successful in accurately simulating golf ball trajectories. The rest of this dissertation will report on the design and development process behind how the Doppler

simulator was upgraded.

1.1 Design Requirements

The focus of this dissertation is to upgrade the AD9 Doppler simulator with suitable electronic hardware. It was stipulated that the upgraded device must be able to achieve the following:

1. Use the existing AD9 Doppler simulator housing, RF board, and antenna.
2. Use the DDS method to generate IQ signals.
3. Match the functionality of the AD9 Doppler simulator.
4. Power and operate the RF hardware to receive, modify and transmit RF signals.
5. Simulate a varying velocity profile that represents golf ball trajectories.
6. The amplitude, phase, and frequency of the Doppler simulator must be adjustable.
7. The Doppler simulator must be portable.
8. The Doppler simulator must be controlled wirelessly.
9. The Doppler simulator must be cheaper than the current DDS processor.

1.2 Functional Specifications

The functional specifications will serve as a guide for the design requirements which is based on the AD9 Doppler simulator. They are the following:

1. Simulated Shot Types: Driver, 6 Iron, and Pitching Wedge

2. Frequency range: 1.5 kHz to 8 kHz
3. Measured velocity profile versus simulated error: ±0.5 m/s
4. IQ mixer sideband suppression: 40 dB
5. Resolution of the DDS phase accumulator: 32-bit
6. Phase resolution: 0.35°
7. DDS look-up table width: 12-bit
8. DDS sampling frequency: Minimum of 20 kHz
9. DAC resolution: 10 - 16 bit
10. Battery usage: up to 10 minutes of continuous use
11. Wireless communication: up to 5m

1.3 Design and Development Methodology

The design and development methodology can be summarized as follows:

1. Performed an analysis of the AD9 Doppler simulator to identify the limitations of the current DDS processor.
2. Researched viable alternatives for the DDS processor and microcontroller that can meet the design requirements.
3. Attained data of live golf shots from a Doppler radar and modeled this data so that it can be simulated using the DDS method.
4. Performed a quantitative simulation of the DDS method.
5. Compared simulated data with measured results.
6. Used open-source simulation software for signal generation and RF design, a combination of Scilab [16] and Octave [17].

7. Used an open-source compiler called the Arduino IDE [18] to code the firmware of the Due Doppler simulator.

8. Used open-source circuit design software called Fritzing [19] to design the peripheral circuits.

1.4 Scope and Limitations

The scope of this dissertation is confined to upgrading the electronic hardware of the AD9 Doppler simulator. Furthermore, its application was evaluated only for golf ball trajectories. No research or development was done on the RF hardware itself nor will it be covered in detail, except were it is relevant to the dissertation. Furthermore, a brief overview of FlightScope Doppler radars is covered in order to give enough background information to aid the design and development.

As stated in the design requirements, only the golf ball trajectory represented by its velocity profile will be simulated. A simple amplitude decay profile was also simulated to represent a realistic profile. The azimuth and elevation angles were not simulated even though these are measured by the radar. This would be beyond the scope of this dissertation and would require further research.

1.5 Executive Summary

- Chapter 2: Literature Review
  This chapter presents a selection of key theoretical topics that were involved in the design and development of the Due Doppler simulator.

- Chapter 3: Feasibility Study
  This chapter provides an argument for the selection of the Arduino Due as a viable microcontroller to upgrade the AD9 Doppler simulator.

- Chapter 4: Design of the Due Doppler Simulator
  This chapter discusses the design approach, the analysis of the AD9 Doppler

simulator, and the design of the Due Doppler simulator.

- Chapter 5: Firmware and Hardware Integration
  This chapter focuses on the firmware and hardware integration based on the design in the previous chapter.

- Chapter 6: Evaluation of the Due Doppler Simulator
  This chapter evaluates the functionality of the device to assess whether it has met the design requirements.

- Chapter 7: Results and Findings
  This chapter provides a summary of the results and findings that was achieved in this dissertation.

- Chapter 8: Conclusions and Recommendations
  This chapter highlights the achieved objectives of this dissertation and comments on further research.

Chapter 2

Literature Review

The purpose of this section is to provide an overview of the literature that formed part of the research and development of this dissertation. This chapter will cover four sections. The first section will provide background information about Doppler radars, the Doppler effect, and the radar range equation. The second section will cover RF frequency synthesis, how RF mixers operate, and the fundamentals behind in-phase and quadrature (IQ) signals. The third section will discuss the principles behind the DDS method and how it functions. The final section will provide an overview of golf ball flight dynamics.

2.1 The Fundamentals of Doppler Radars

A standard Doppler radar transmits an RF signal into free space and measures signal reflections from its environment. Moving objects interact with the RF signal and reflect it back to the radar as an echo. This reflected signal has a phase difference relative to the transmitted signal. The Doppler frequency is the rate of change of this phase. Furthermore, the Doppler frequency is used to determine the radial velocity of an object as well its orientation towards or away from the radar [20]. Objects moving away from the radar have a lower Doppler frequency compared to objects moving towards the radar. This effect can be seen

in Figure 2.1 which shows how a golf ball interacts with electromagnetic waves.

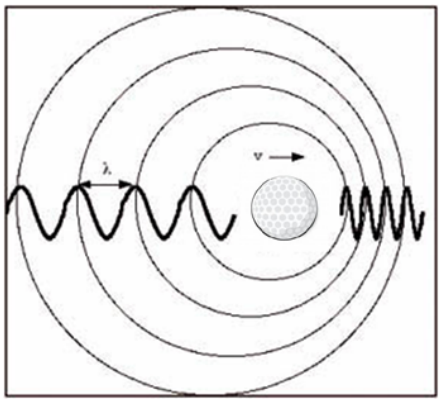


Figure 2.1: A representation of how the Doppler frequency changes whether it is moving towards or away from the device measuring it [21].

It can be observed that if the ball is moving towards the radar it appears that the ball is compressing the waves in front of the ball. This results in a shorter wavelength and therefore a higher Doppler frequency. The converse is also true, if the ball is moving away from the radar the wavelength is longer and hence has a lower Doppler frequency. In order to determine the radial velocity of projectiles, the Doppler frequency can be related to the radial velocity using the following equation:

$$v_r = \frac{\lambda f_d}{2}. \tag{2.1}$$

Where v_r is the radial velocity, λ the wavelength, and f_d the Doppler frequency [20]. The wavelength is determined from the following equation:

$$\lambda = \frac{c}{f_{RF}}. \tag{2.2}$$

Where c is the speed of light and f_{RF} is the transmitted RF signal [20]. The amplitude of the signal is a measure of the returned signal strength. As the target moves away from the radar, the radar will measure its Doppler frequency. However, in some cases, it won't be able to measure the target along its full

trajectory as the received signal power from the target reduces at a rate of $\frac{1}{R^4}$, where R is the range to the target relative to the radar [20]. Furthermore, the signal amplitude decays to below the noise floor of the system which makes it undetectable. This can be further observed by the ideal radar range equation [20] which categorizes the antenna and target parameters:

$$R = \sqrt[4]{\frac{P_t G_t G_r \lambda^2 \sigma}{P_r (4\pi)^3}}. \tag{2.3}$$

- P_t = transmitted power $[W]$
- P_r = received power $[W]$
- G_t = transmitter gain
- G_r = receiver gain
- σ = radar cross section (RCS) $[m^2]$
- λ = wavelength $[m]$

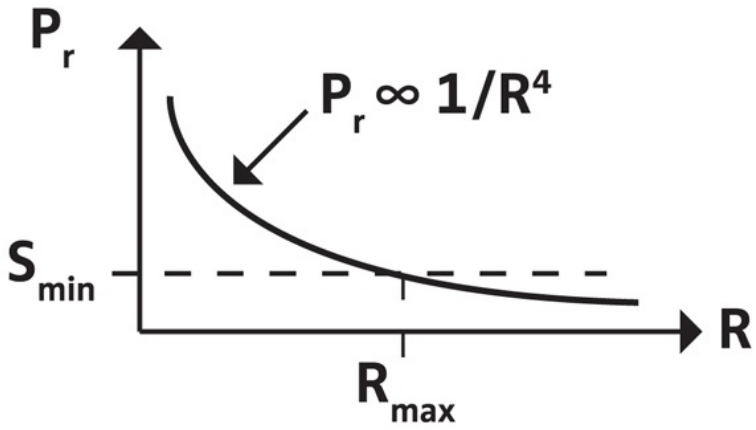


Figure 2.2: The relationship between the range and signal power received by a target moving away from the radar [22].

The signal power versus range is further represented in Figure 2.2. Where S_{min} is defined as the minimum signal power received. These two fundamental properties of Doppler radars are key features to this dissertation.

2.2 RF Frequency Synthesis

RF frequency synthesis is a fundamental building block of radars. It covers a wide range of techniques such as frequency multiplication and division, DDS, frequency mixing, and phase-locked loops. One of the techniques that is fundamental to this dissertation is frequency mixing via an RF mixer.

2.2.1 RF Mixer

The function of an RF mixer is to multiple two input ports or signals together which produces an output signal [23]. By convention, the two input ports are called the radio frequency (RF) and local oscillator (LO) ports respectively. The output port is called the intermediate frequency (IF). This arrangement is called the down-conversion method (DCM), this means that a higher RF input frequency is changed to a lower baseband frequency [24]. The RF input signal can be represented by the equations below:

$$f_{RF} = A\sin(w_1 t), \tag{2.4}$$

$$f_{LO} = B\sin(w_2 t). \tag{2.5}$$

Where ω is represented as $2\pi f_0$ and f_0 is the carrier frequency. Then by multiplying Equation 2.4 and 2.5 the output of the IF signal can be shown as:

$$f_{IF} = AB\sin(w_1 t)\sin(w_2 t). \tag{2.6}$$

Equation 2.6 can be expanded as a product-sum trigonometric [25] identity defined as:

$$f_{IF} = AB\sin(w_1 t)\sin(w_2 t) = AB(\frac{\cos(w_1 t - w_2 t)}{2} - \frac{\cos(w_1 t + w_2 t)}{2}). \quad (2.7)$$

So the output of the mixer can be represented by two frequencies as shown below:

$$f_{IF_1} = f_{RF} + f_{LO}, \quad (2.8)$$

$$f_{IF_2} = f_{RF} - f_{LO}. \quad (2.9)$$

As mentioned before, this configuration of having the IF as the output of the RF mixer is known as the down-conversion method (DCM). Another format is to interchange the RF and IF signals. This is known as the up-conversion method (UCM) [24]. So the new configuration becomes the following:

$$f_{RF_1} = f_{LO} + f_{IF}, \quad (2.10)$$

$$f_{RF_2} = f_{LO} - f_{IF}. \quad (2.11)$$

For the purpose of this dissertation, the focus will be on the UCM. The outputs f_{RF_1} and f_{RF_1} are known as the upper-sideband (USB) and lower-sideband (LSB) signals respectively [24]. As mentioned earlier, an object moving towards the radar will have a higher Doppler frequency and can be observed as the USB. The object moving away from the radar will have a lower Doppler frequency which can be observed as the LSB. Since golf balls are measured moving away from the radar, the USB needs to be suppressed and the LSB amplified to improve tracking on this sideband. One way to achieve this is to use a single-sideband (SSB) RF mixer [24]. This type of RF mixer uses phase cancellation coupled with complex IQ processing to resolve the USB and LSB [24]. In order to do this, the real part of the LO and IF signals need to be converted to complex IQ signals that have a 90° phase offset. These signals are still analog at this point and therefore are real. In the first case, a SSB RF mixer has an internal quadrature phase shifter that generates two complex IQ signals from the LO

signal to produce the following:

$$LO_I = \sin(w_1 t), \tag{2.12}$$

$$LO_Q = \cos(w_1 t). \tag{2.13}$$

Furthermore, the IQ part of the IF signal is generated digitally by the DDS method through two DACs. They are then represented as:

$$IF_I = \sin(w_2 t), \tag{2.14}$$

$$IF_Q = \cos(w_2 t). \tag{2.15}$$

The SSB RF mixer then mixes these two IQ signal pairs. A block diagram of an IQ RF mixer can be seen in Figure 2.3. The LSB is required so the two I signals

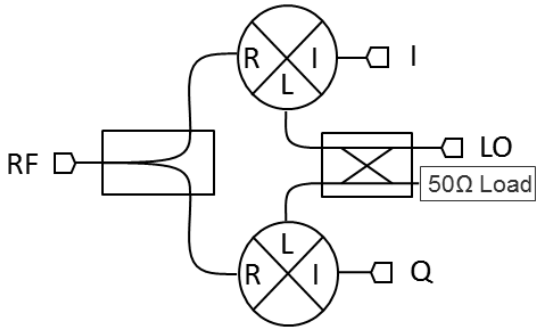


Figure 2.3: A block diagram of an RF upconversion mixer [24].

and the two Q signal pairs are mixed. The output signal of the SSB RF mixer can therefore be represented by the following:

$$\begin{aligned} RF &= LO_I \times IF_I + LO_Q \times IF_Q \\ &= \frac{(\cos(w_1 - w_2)t)}{2} - \frac{(\cos(w_1 + w_2)t)}{2} \\ &+ \frac{(\cos(w_1 - w_2)t)}{2} + \frac{(\cos(w_1 + w_2)t)}{2}. \end{aligned}$$

The two LSB signals are added and the two USB signals are cancelled since they have opposite signs. The resulting RF signal can be represented by the following:

$$RF = (cos(w_1 - w_2)t). \tag{2.16}$$

This process can be observed in Figure 2.4.

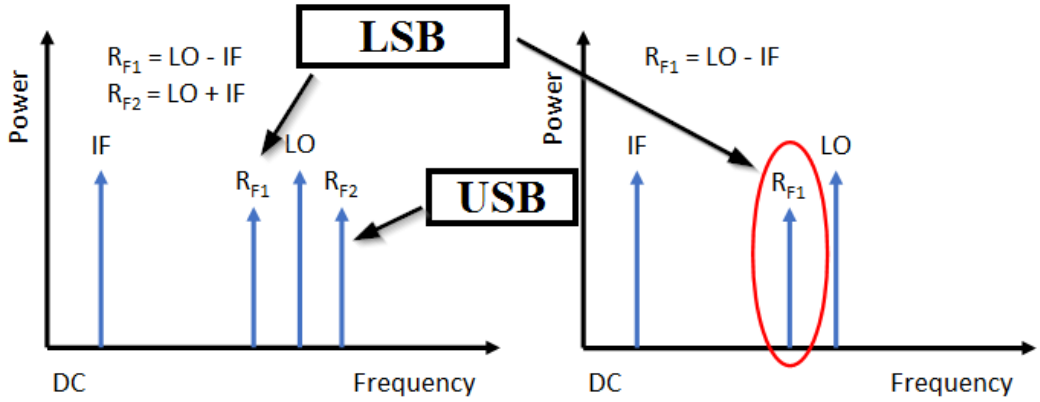


Figure 2.4: A representation of the output of the RF mixer showing the lower and upper sidebands R_{F1} and R_{F2} around the carrier frequency LO. Through the SSB RF mixer the USB, R_{F2}, is suppressed as shown on right.

This is the ideal case. When RF signals are mixed in practice, intermodulation products are formed due to the nonlinearity of the mixer. This effect is seen by frequency multiples of the actual frequencies [26].

2.2.2 In-Phase and Quadrature Signals

A fundamental feature of frequency synthesis and determining the direction of a target is dependent on IQ signals. The definition of IQ signals are two identical waveforms that have a 90° phase offset relative each other. This can be seen in Figure 2.5.

It can be observed that I is leading Q. Through the UCM configuration of an SSB RF mixer, this would produce an RF frequency slightly below the LO as the opposite complex pairs would be multiplied together by the SSB RF mixer

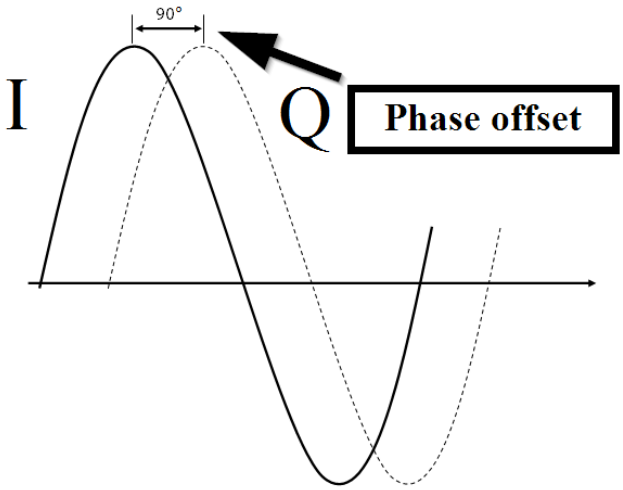


Figure 2.5: A representation of the I and Q waveforms showing how I is leading Q.

to generate the LSB. Similarly, if Q was leading I the RF frequency would be slightly above the LO and would then generate the USB. Furthermore, these waveforms can be represented as follows:

$$y = I\sin(wt) + Q\cos(wt). \tag{2.17}$$

IQ signals form a fundamental part of RF frequency synthesis and the DDS method. This shall be seen later in this dissertation.

2.3 The Principles of DDS

The term DDS refers to either a method of direct digital synthesis or a device known as a direct digital synthesizer. Both terms refer to generating analog signals using digital techniques [28]. This dissertation will primarily refer to DDS as a method, unless otherwise stated. The DDS method consists of a reference oscillator, a numerically controlled oscillator (NCO), and a digital-to-analog converter (DAC) [29]. The reference oscillator provides the frequency

accuracy of the DDS and acts as the clock to the NCO. The NCO uses the clock to generate a quantized version of a digital waveform. The period of this waveform is controlled by a tuning frequency. The quantized waveform is then converted by the DAC to an analog signal. The term "waveform", is used to refer to a theoretical or digital representation of periodic function whereas a "signal" refers to an analog periodic function. The NCO itself can be divided into three more parts which can be seen in Figure 2.6. The first part involves generating a waveform look-up table (LUT) [30]. This LUT is an array in memory that contains the amplitude values of a template waveform. The second part is a phase accumulator (PA) which is a high resolution counter. Its output is used as the index value for the LUT. The final part of the NCO is the tuning word or phase increment (PI). This is the ratio of the desired frequency and reference clock of the embedded hardware used to tune the PA.

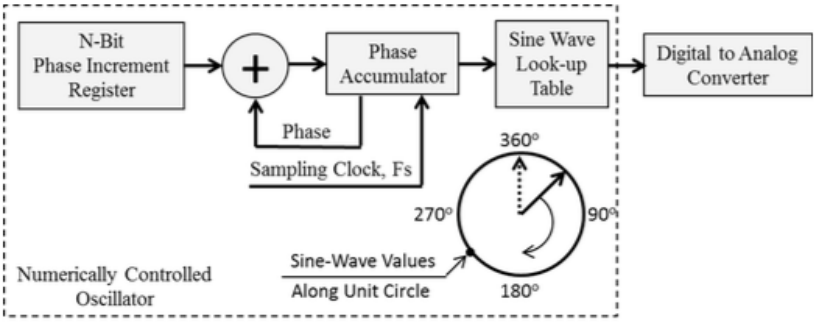


Figure 2.6: A flow diagram of the DDS method showing the NCO and DAC operation [31].

2.3.1 Look-Up Table

The LUT is a table of amplitude values representing a waveform. The size of the table and waveform determines the final output of the DDS method. In most cases a sine or cosine waveform is used although square waveforms can also be generated. The length of the table (l_{LUT}) determines the phase resolution of the

generated waveforms which can be determine by using the following equation:

$$\phi_{res} = \frac{2\pi}{l_{\mathrm{LUT}}}. \tag{2.18}$$

The width of the table (w_{LUT}) determines the amplitude resolution which is usually designed to fit the voltage range of the DAC (V_{DAC}). This is calculated by the following equation:

$$A_{res} = \frac{V_{\mathrm{DAC}}}{w_{\mathrm{LUT}}}. \tag{2.19}$$

2.3.2 Phase Accumulator

The phase accumulator consists of an n-bit adder and tuning register, which generates an n-bit value on each new clock-cycle. Each output consists of the previous output added with the PI. Over a successive amount of clock-cycles the output forms a staircase shaped waveform. This is an accumulating value that is associated by a certain step size, gradient and cutoff point. The PA requires a high precision bit resolution to achieve a fine frequency resolution which is usually achieved with floating point arithmetic. However, for embedded processors, using floating point arithmetic requires many instructions which can cause a bottleneck for the DDS method. Thus, an alternative procedure is to use fixed point arithmetic as a simple way of representing the same precision as floating point numbers. This is done by scaling the PA value as a large n-bit value and then performing finite increments by overflowing into successive bits. The final result of the PA is to provide an index value for the LUT. The LUT and output of the PA needs to match in terms of bit size. However, the PA values is usually a lot larger than the LUT as it needs to use fixed point arithmetic [32][33]. To cater for this, the PA value has its lower bits truncated in order to match the LUT. Figure 2.7 shows a phase wheel representation of the PA and how through each successive phase step it accesses the LUT to generate the waveform.

Lastly, the frequency resolution (f_{res}) of the PA can be determined by using the following equation. Where f_s is the sampling frequency and N is the resolution

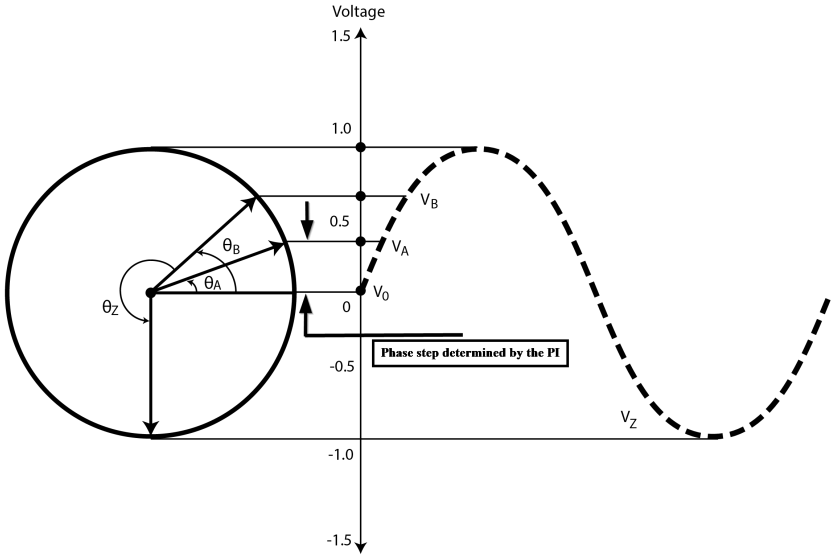


Figure 2.7: A representation of the phase accumulator as a phase wheel and how it generates a sine waveform [29].

of the PA:

$$f_{res} = \frac{f_s}{2^N}. \tag{2.20}$$

2.3.3 Phase Incrementor

The PI is one of the inputs to the PA and is the ratio of the tuning frequency and reference oscillator, it also determines the slope of the PA. This is done through the following equation where f_t is the tuning frequency:

$$PI = f_t \frac{2^N}{f_s}. \tag{2.21}$$

The PI also needs to be scaled to the same number of bits used by the PA.

2.3.4 Digital-To-Analog Converter

A DAC converts digital waveforms to analog signals by assigning a voltage proportional to the discrete numerical value that it is given. The rate at which the DAC assigns these voltages is the sampling frequency of the DAC where the maximum sampling frequency is usually in the megahertz range [34]. According to Nyquist's sampling theorem [35], for the DAC to reconstruct these waveforms as analog signals, the sampling frequency needs to be at least twice that of the highest frequency of the reconstructed waveform. Furthermore, the output of the DAC has a quantized profile. In order to smooth this analog signal a low-pass filter (LPF) is used.

2.4 Golf Ball Flight Dynamics

This section aims to provide a general overview of the sport of golf, its terminology, and golf ball flight dynamics that are relevant to this dissertation.

2.4.1 Golf Terminology

There are various terms that are unique to golf. Since the application of the Doppler simulator is for golf, these terms will be used where it is necessary.

- Clubs: A variety of different clubs are used in golf. Each with different shaft length, club head size, and face angle. The three main types of clubs that will be discussed are a Driver, a 6 Iron, and a Pitching Wedge. These will be used to refer to the type of ball that was hit. For example, a Driver is used to hit the ball the furthest, with the highest velocity, and lowest launch angle. A Pitching Wedge is used to hit the ball the shortest distance, with the lowest velocity, and highest launch angle. A 6 Iron is in between a Driver and a Pitching Wedge [36].
- Impact: The moment the player strikes the ball with the club.

- Ball Flight: The path of the ball after impact while it is in the air. This may be referred to as a "shot" as well. This does not include the first bounce or what happens thereafter, that is known as the "roll" [36].
- Carry: This is the total distance or range the ball has covered during its ball flight.
- Loft: This is the angle the club face makes with the shaft. The more loft the club has the larger the launch angle. For example, a Driver may have a 2° loft while a Pitching Wedge may have a 45° loft [36].

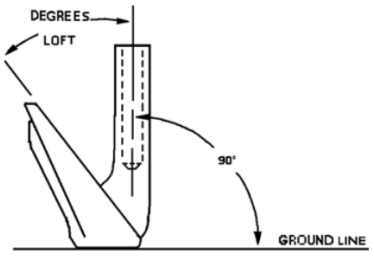


Figure 2.8: The shaft loft [37].

- Spin: This is the rotation of the golf ball around its axis which is usually given as backspin. A Driver may produce a spin of 2000 revolutions per minute (RPM) while a Pitching Wedge may produce a spin of 8000 RPM [37].

Table 2.1 shows some typical metrics that can be expected from professional golfers for Driver shots.

Table 2.1: A table of metrics that can be expected from professional golfers [37].

Player	**V (m/s)**	**Launch Angle (deg)**	**Spin (rpm)**	**Carry (m)**
Vijay Singh	78.68	10.7	2600	282.46
Robert Allenby	71.97	8.5	2390	274.59
Peter Lonard	75.10	11.7	2673	268.38
Phil Mickelson	79.57	13.0	2200	281.64
Ernie Els	77.78	11.5	2400	292.24

2.4.2 Flight Dynamics

The primary factors that effect the golf ball trajectory are the launch velocity, spin, drag, and launch angle. These are closely related to the club that is being used to strike the golf ball. For example, the loft of a club may cause a higher spin and launch angle, while the swing speed of the club will have a significant effect on the launch speed [37]. The effect can be visualized in Figure 2.9. It shows the relationship between the velocity profile and the height relative to the range using a generic golf model [38]. This model uses the initial launch velocity, spin, drag, and launch angle to generate the trajectory. It can be observed that the velocity profile has a consistent decay profile, then at the end of the velocity profile there is a slight increase in velocity. This is due to the ball accelerating after it has reached its apex. This effect shall be seen later with data captured from the Doppler radar.

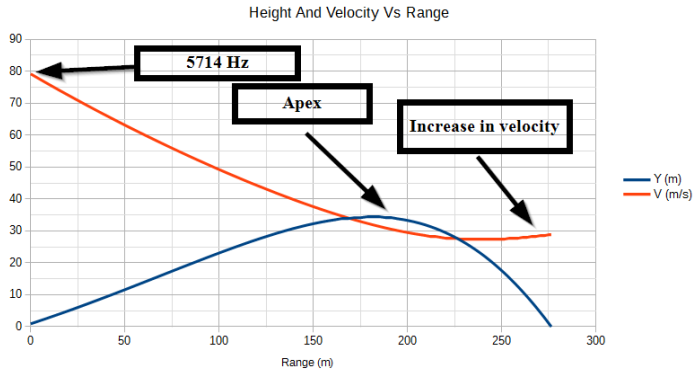


Figure 2.9: This figure shows the representation of the velocity and height profile versus range of a golf ball in flight.

Chapter 3

Feasibility Study

This section will assess the value of upgrading the AD9 Doppler simulator. As mentioned in the design requirements, the main purpose of the upgrade is to have a new Doppler simulator that would be able to simulate the velocity profile of a golf ball trajectory. This section aims to focus on the main limitations of the AD9 Doppler simulator and then provides possible solutions to meet the design requirements.

3.1 Doppler Simulator Solutions

The AD9 Doppler simulator uses a AD9854 DDS CMOS processor [39]. This is a dedicated DDS device that was designed to generate digital waveforms and then convert them to analog signals. However, the AD9854 is limited in its control over its two IQ DAC outputs. It allows for amplitude and frequency modulation (FM), with phase shift keying, and other features like FM chirp functionality. However, it does not cater for variable phase modulation since the electronic DDS has a fixed phase offset of 90°. The purpose of the AD9 Doppler simulator was to simulate a constant velocity of 3.5 kHz at varying amplitudes. In theory, it is capable of producing a varying velocity profile matching that of a golf ball trajectory. However, due to newer and cheaper technology, other

electronic hardware was investigated. As discussed in the literature review, the DDS method is a numerical process that uses a PA and LUT to generate a variety of waveforms. These digital waveforms are then processed by two DACs to generate quadrature IQ signals. Therefore, the DDS method and two dedicated DACs are critical to meeting the design requirements.

In electronic prototyping, the Arduino series of development boards has been used by hobbyists for a variety of projects [40] [41]. Once such area is that of frequency synthesis. A variety of Arduino projects have been developed that use the Arduino Uno and the DDS method as a simplistic sinewave generator [42] and a basic audio player [43]. There are also dedicated Arduino shields that use the AD9851 DDS CMOS processor, similar to the AD9854, to generate high frequency waveforms that can be integrated with the Arduino development boards [44].

Taking into consideration the budget for the upgrade, the Arduino Uno was considered as the first choice since they use cheap 8-bit 16 Mhz Atmel processors. The main problem with these microcontrollers is that they have no dedicated DACs as part of the processor architecture. However, there are two solutions to this problem. The first solution is to generate a DAC through its dedicated PWM pin, although this only give an 8-bit DAC resolution. The second solution would be to use an Arduino peripheral board called "breakout boards" that can be attached to the Arduino Uno. Such a board is the MCP4725 which has a dedicated 12-bit DAC that is operated by the I2C bus of the Arduino [45]. However, this would require further components along side the Arduino Uno. Further research into the Arduino library of projects showed a simple waveform generator using the Arduino Due microcontoller [15]. This is a unique Arduino board that uses an Atmel Arm Core 84 MHz processor which has two dedicated 12 bit DACs. Furthermore, the Arduino Due has been used in simple stereo audio player applications using the DDS method [46][32]. This means that each DAC could also serve as IQ signals. These boards are also cheap and available in large quantity. However, the true comparison would be between the Arduino Due's microntroller, the SAM3X8E [47], and the AD9854 DDS procssor used in the AD9 Doppler simulator.

3.2 Processor Comparison

The Arduino Due uses a 32-bit ARM core processor called the SAM3X8E. It is has the fastest processor in the Arduino series with a clock rate of 84 MHz. Most importantly it has two internal 12-bit DACs so no extra breakout boards are required. Furthermore, this Arduino is a multi-purpose microcontroller which includes a variety of I/O pins and a USB communication interface. As mentioned earlier, the Arduino Due has been used before in DDS and frequency synthesis applications. A comparison of the Arduino Due's microcontroller, the Atmel SAM3X8E, and the AD9854 processor is shown below:

Table 3.1: A comparison of the two DDS processors

	AD9854	**SAM3X8E**
Parameter		
Clock Rate (MHz)	300	84
Integrated DAC	2	2
DAC Resolution (bit)	12	12
DAC Sample Rate (MSps)	300	2
Tuning Frequency (bit)	48	32
Output Voltage (V)	3.3	3.3
RAM (kB)	NA	96
Flash Memory (kiB)	NA	512
Operating Temperature (°C)	-40 to 85	-40 to 85
Price (R) [48][49]	$\sim$ 1500	$\sim$ 270

One can see that the AD9854 out performs the SAM3X8E on most of the parameters, excluding the price. The actual price of the complete Arduino Due board is approximately R450 [50] which is also less than the AD9854 processor. It is important to note that the AD9854 is a dedicated DDS processor and has been designed specifically for high-end synthesis, processing and communication. This can be seen with the large difference in the DAC sample rates. This is because the AD9854 has an electronic DDS component. Whereas the Arduino Due uses a firmware based DDS that is limited by the interrupt routines of the timers. However, this allows for a unique feature of the SAM3X8E. That is, it is able to simulate multiple targets moving towards or away since it can generate multiple

DDSs. In summary, the Arduino Due with its SAM3X8E microcontroller does meet the design requirements according to its specifications. Furthermore, the upgrade is more flexible and relies on open-source platforms. This allows for additional functionality to be incorporated in the future, should the need arise.

The proposed Arduino development board chosen for the upgrade was the Arduino Due. The main reasons was because of its two internal 12 bit DACs, price, and usage in frequency synthesis applications. Furthermore, this is a single board that requires no further build time or electronic hardware to function, whereas the AD9854 is only a processor. The rest of this study will investigate the feasibility of using the Arduino Due to upgrade the AD9 Doppler simulator. The upgraded AD9 Doppler simulator will further be known as the Due Doppler simulator in the rest of this dissertation.

3.3 Chapter Summary

This chapter aimed to propose the Arduino Due as a suitable upgrade for the AD9 Doppler simulator. It showed that the Arduino Due has implemented the DDS method before with its two DACs. It also showed a comparison between the two processors and showed that although the SAM3X8E is a low-end processor compared to the AD9854, it can still meet the design requirements stated in Chapter 1.

Chapter 4

Design of the Due Doppler Simulator

This chapter looks at the design of the Due Doppler simulator. There are three main areas of design in this dissertation. These are electronic hardware, RF design, and signal processing. This chapter begins by analyzing the features of the AD9 Doppler simulator and identifies how it operates. It then discusses how the AD9 Doppler simulator can be upgraded to meet the design requirements. Thereafter, the velocity flight models are derived using empirical data measured by a Doppler radar. Lastly, the design of the firmware and electronic hardware is discussed.

4.1 System Design

The Doppler simulator is a system of electronic components that is integrated with firmware to simulate a velocity profile. As discussed in the feasibility study, the Arduino Due was chosen as a suitable upgrade for the AD9 Doppler simulator. In order to design the upgrade, the dissertation will follow a design cycle process that consists of analysis, synthesis, and evaluation. That entails analysing the existing Doppler simulator to see how it operates. Then through synthesis, design

and integrate the Arduino Due into the existing Doppler simulator. And finally, evaluate the upgraded device to assess whether or not it has met the design requirements. These three parts of the design process will be covered in the following chapters.

4.2 Analysis of the AD9 Doppler Simulator

The first part of the design process is to analyze the existing Doppler simulator and assess how it should be upgraded. The section that follows will analyze the overall operation and functionality of the AD9 Doppler simulator.

The AD9 Doppler simulator is a passive device that was designed to simulate a constant velocity profile. A top view of this device is shown in Figure 4.1.

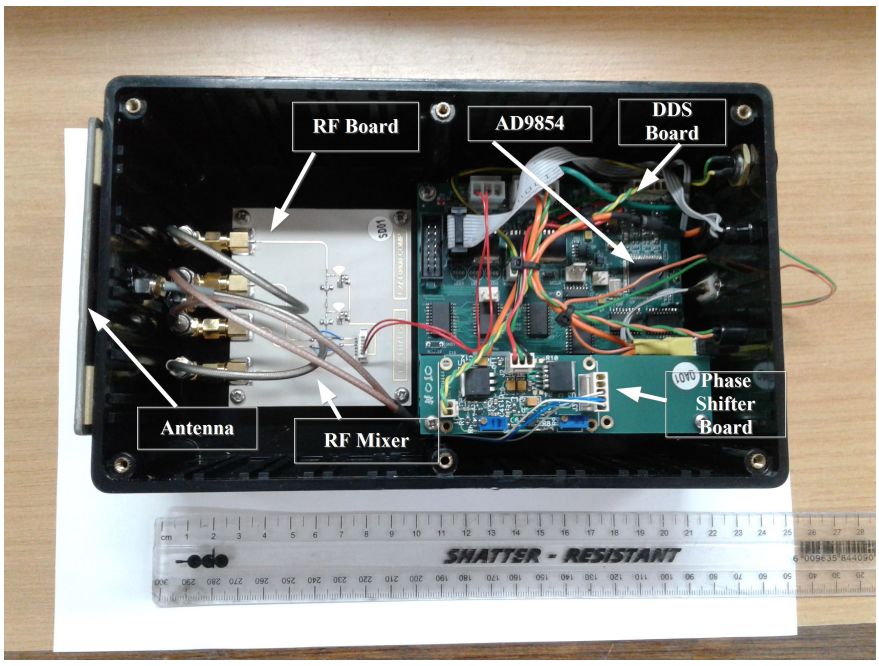


Figure 4.1: A top view of the AD9 Doppler simulator

It is operated in conjunction with a Doppler radar which transmits a carrier signal that is received by the antenna of the Doppler simulator. This carrier signal is then modified with a Doppler frequency and is transmitted back to the

Doppler radar. The Doppler radar then interprets the Doppler frequency as a moving target relative to the radar. This data is then downloaded to a PC and analyzed. An overview of this is shown in Figure 4.2.

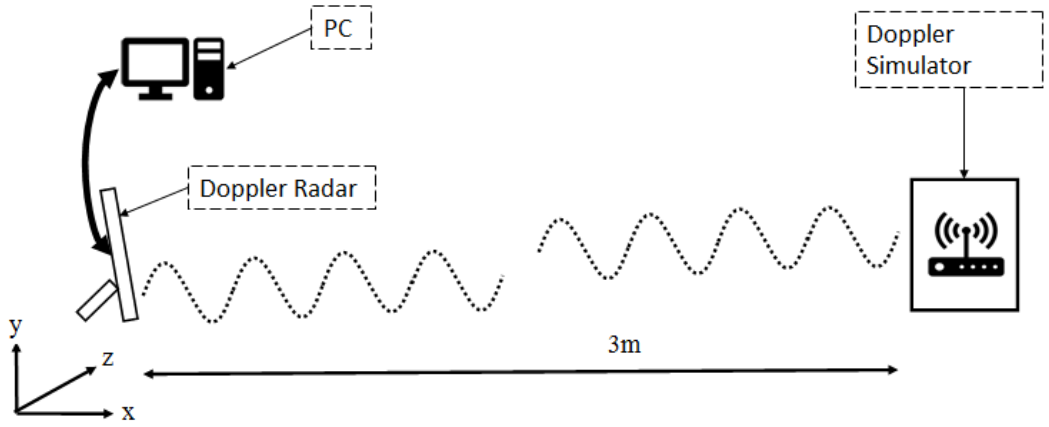


Figure 4.2: A layout of how the current AD9 Doppler simulator is used relative to a Doppler radar.

The main components of the AD9 Doppler simulator are the following:

- Antenna: X-band 4x2 patch antenna that receives and transmits RF signals
- RF Board: Comprises of an SSB RF mixer, the HMC521 [51], and an NBB-300 amplifier [52]
- Circulator: Allows for sharing of a single antenna for receive and transmit
- DDS Processor: The AD9854 CMOS DDS processor
- DDS Board: The board that operate the DDS processor and powers the RF board. It also serves as a communication link to program the DDS processor
- Phase Shifter: This board shifts the phase between the I and Q signals to compensate for RF mixer offsets
- Housing: Contains all the components of the Doppler simulator

The SSB RF mixer, shown in Figure 4.3, requires two input signals called the LO and IF, which generates an output signal called the RF. These IF signals are generated by the DDS board which is made up of the AD9854 processor, peripheral components, and a phase shifter board. The purpose of the AD9854 DDS processor is to generate quadrature IQ signals. These signals are fed to a LPF and blocking capacitor and then fed to the phase shifter board which is used to tune the phase of the IQ signals to compensate for RF mixing offsets. The peripheral components provides power and acts as the communication link to program and control the DDS processor. The device is powered with 12V DC and communication is via an RS232 connection. The housing houses all these components with the antenna placed on the outside of the device. These components are shown in Figure 4.4.

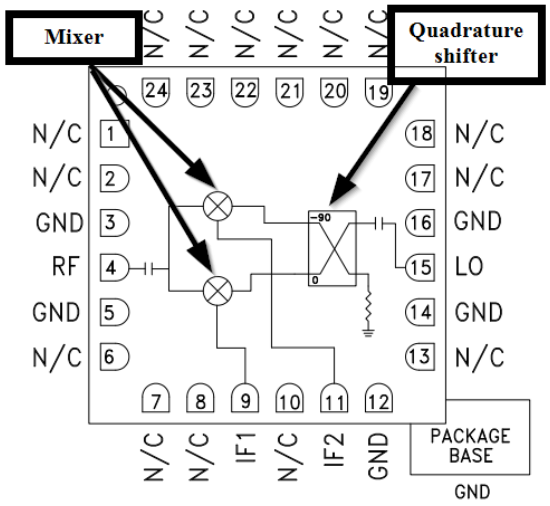


Figure 4.3: An image of the "HMC521" RF mixer showing the input pins of LO and IF1 and IF2. As well as showing the RF output pin.

The firmware operation of the device has two states. It is either simulating or on standby. When it is simulating, it simulates a fixed frequency of 3.5 kHz. The only variable that can be controlled is the amplitude. The firmware flow diagram of the device can be seen in Figure 4.5.

Lastly, the top view of the RF board is shown in Figure 4.6

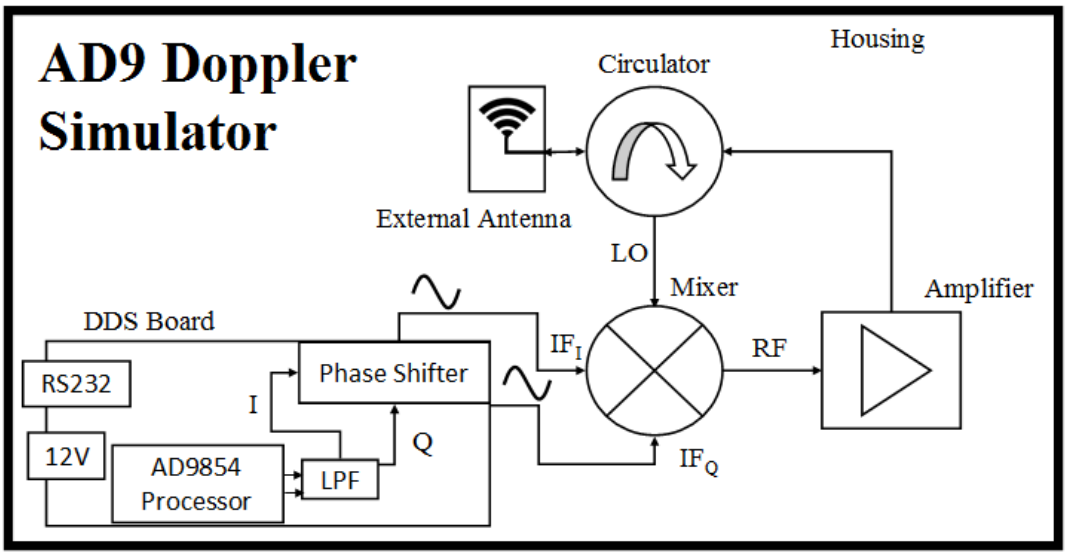


Figure 4.4: A block diagram showing the main components of AD9 Doppler simulator.

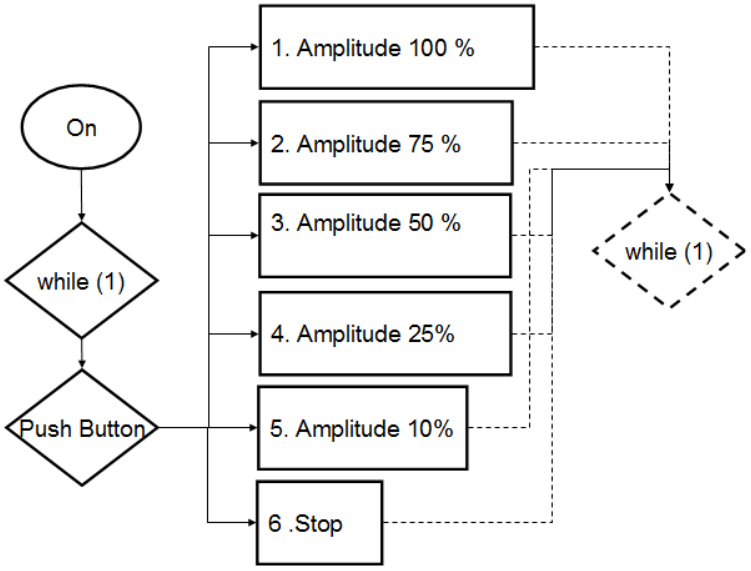


Figure 4.5: A functional flow diagram of how the AD9 Doppler simulator is operated and showing its various modes.

Appendix C shows the outputs generated by the AD9 Doppler simulator. Further details about the device can be found in Appendix B and C. The next section will begin the discussion about upgrading the Doppler simulator.

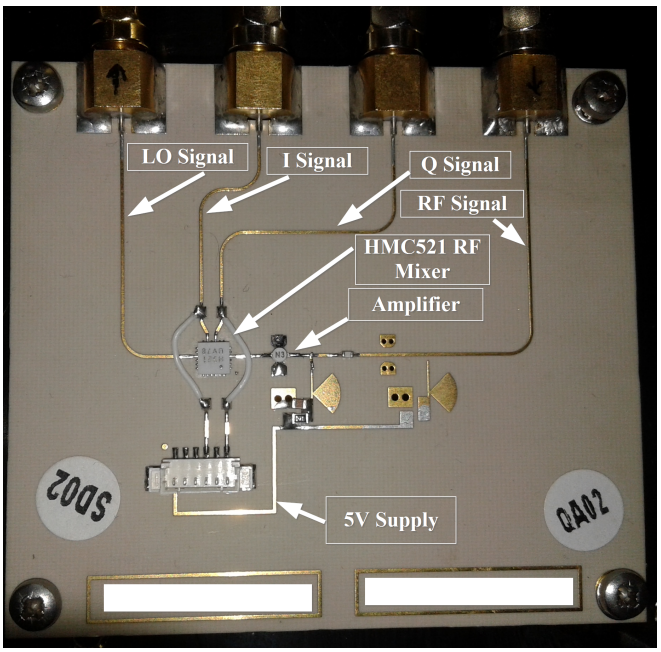


Figure 4.6: A top view of the RF board.

4.3 Synthesis of the Due Doppler Simulator

This section will discuss the design behind upgrading the AD9 Doppler simulator to the Due Doppler simulator. Since the RF hardware functionally performs the task of simulating the Doppler signal, the only part of the AD9 Doppler simulator that needs to be replaced is the DDS board.

4.3.1 System Operation

The operation of the Due Doppler simulator will follow the same general structure as the AD9 Doppler simulator. The only difference will be, that it will be able to generate varying velocity profiles. Using the block diagram of the AD9 Doppler simulator as a template, the general structure of the Due Doppler simulator is shown in Figure 4.7.

The first phase of the design is to generate the velocity profile. To do this, the characteristics of the velocity profile for various golf balls in flight need to be

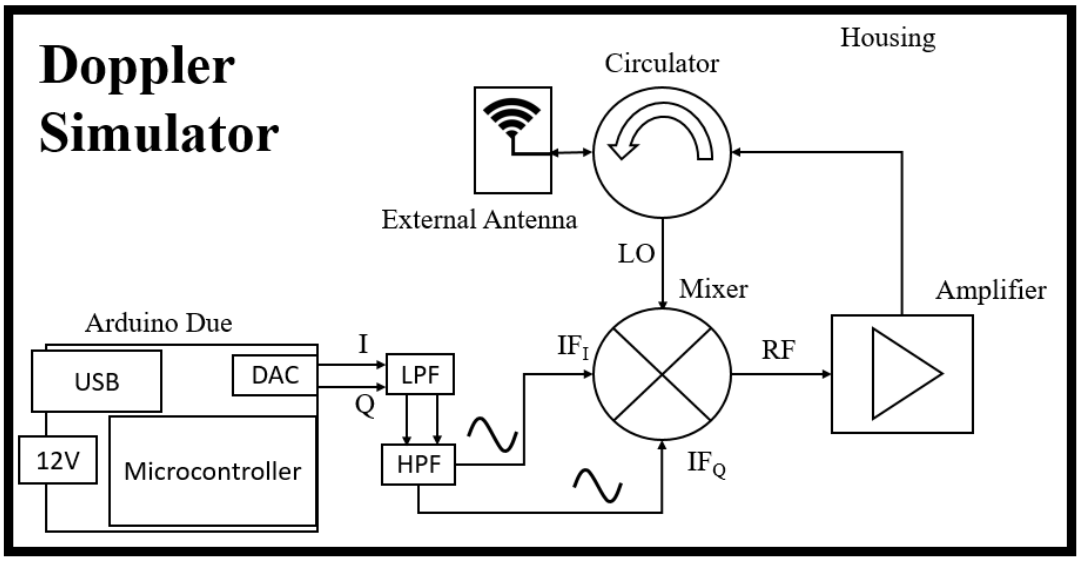


Figure 4.7: Flow diagram of the Due Doppler simulator function.

determined.

4.4 Flight Model Design

The literature review briefly discussed golf ball trajectories. This section will take an empirical approach to designing the flight models. It will take golf shots measured by the Doppler radar and fit polynomials to the actual velocity profile. From this, a flight model will be derived and then simulated by the Doppler simulator.

4.4.1 Empirical Simulation

The literature review briefly discussed golf ball flight dynamics. It also showed how the velocity profile was compared to the range profile. By using an empirical approach, the velocity profile can be extracted from the measured golf ball flight data. This was done by recording data from a Doppler radar for three different types of golf shots. These are defined by the types of clubs that are used, namely

a Driver, a 6 Iron, and a Pitching Wedge.

4.4.2 Velocity Profile

In Figure 4.8 the measured raw Doppler data of a 6 Iron golf shot from one of the channels of a Doppler radar is shown. The figure shows the sampled voltage level amplitude versus time.

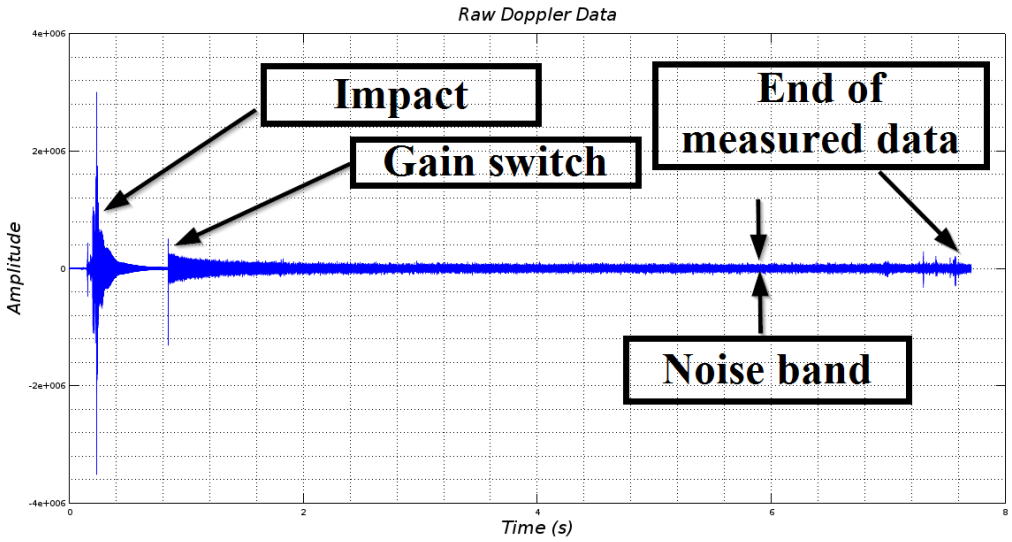


Figure 4.8: Raw Doppler data of a 6 Iron golf shot.

One can observe that there is an initial burst of energy just before 500ms, this represents the impact point of the club with the ball. Thereafter, there is a sudden decrease in energy. Just before one second a gain switch occurs which is used to amplify the signal, as the signal of the ball gets weaker as it travels further away from the radar. The literature review showed the relationship between the Doppler frequency and the radial velocity of a moving object. However, since the data was captured in the time domain, the change in frequency and hence the change in velocity cannot be derived easily. In order to evaluate the Doppler frequency the Doppler data from the time domain needs to be transformed into the frequency domain by using the fast-Fourier transform (FFT) method. However, this would only provide the frequency spectral content over

a certain period of data. To cater for the changing frequencies, a spectrogram needs to be implemented. A spectrogram is implemented by doing short-time FFTs that overlap with each other over the entire data set. To determine the velocity profile from a spectrogram plot, the largest peak in each short-time FFT segment is determined and then this is plotted to represent the velocity profile [53]. The function "specgram" in Octave was used to perform the spectrogram. This function also requires certain parameters to implement a spectrogram [54]. Each FFT is regarded as segment that makes up the entire spectrum. Choosing an appropriate size of the segment will determine the resolution of the frequency spectrum. Before an FFT is done on a segment, it is windowed using a hanning window by default. The choice of the window is important consideration as it determines the time-frequency resolution [53].A gradual window may show more harmonic features whereas a steeper window averages these features to produce the most dominant frequency. Between each succesive FFT there is a step size which controls the time scale. To gain better interpolation between frequency points, the FFT length can be set to be larger than the window length. Another parameter is the overlap which specifies the number of samples to overlap between successive segments [54]. The final parameter is the frequency resolution. This is specified by the sampling rate that the samples were captured at. A summary of the spectrogram parameters can be seen in Table 4.1. Using these

Table 4.1: Spectrogram Parameters

	Parameter	**Value**
1	FFT Segment Size	1024
2	Sampling Frequency	17045 Hz
3	Window Type	Hanning
4	Window Length	1024
5	Overlap	512

parameters a spectrogram of the 6 Iron Doppler data was performed, this resulting spectrogram plot is shown in Figure 4.9. It can be observed that a decreasing frequency profile begins at around 200 ms and ends at about 5 s. It can also be observed that there is a steep increase in frequency at 200 ms. This frequency profile relates to the golf club velocity profile. Lastly, the signal amplitude of the

signal becomes too weak after 4.5 s and is eventually lost.

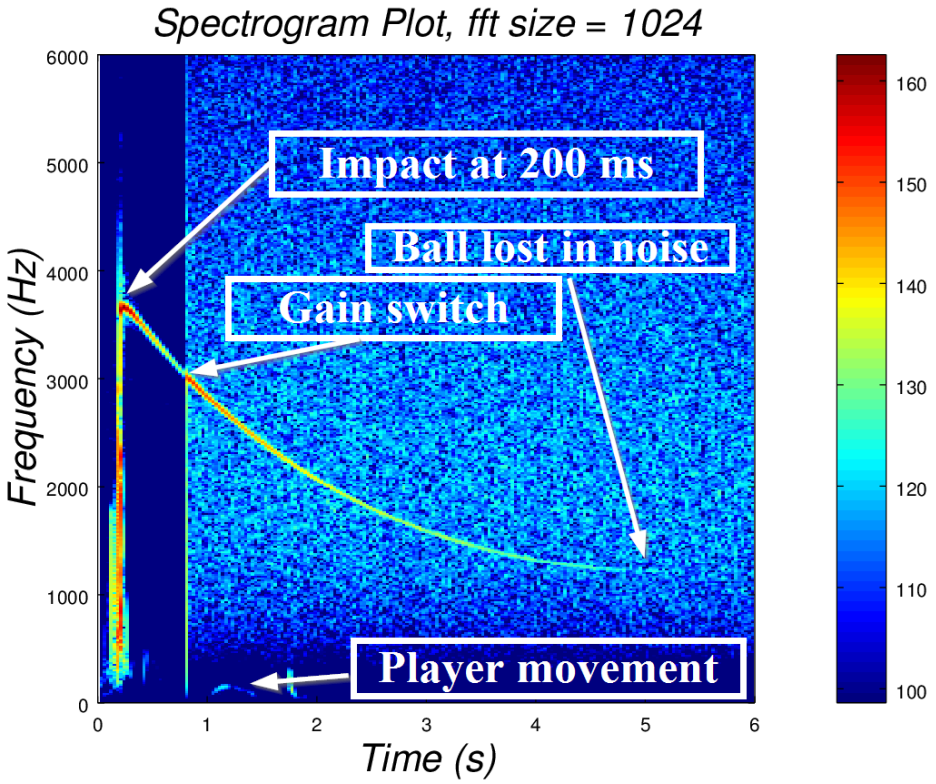


Figure 4.9: A spectrogram plot of the raw Doppler data of a 6 Iron golf shot.

The purpose of doing a spectrogram plot is to be able to display the velocity profile of a golf ball in flight. Thereafter, the largest peaks were tracked in each segment and then a polynomial line was fit to this profile. This became the flight model for each golf shot which excludes the profile before impact. However, most of the ball is in a region where there is a lot of noise and therefore it may track on spurious points. To normalize the noise, the average noise value for a segment was determined in that region and then subtracted from all the segments, hence normalizing the entire spectrum. A second order polynomial was chosen to model the velocity profile as this represented each velocity profile closely. An exponential fit was also attempted, although it did not fit the slight acceleration of the golf ball at the end of the profile. The polynomial coefficients were determined by using the "polyfit" function in Octave [55]. This function does

a polynomial interpolation of the input data and returns coefficients that minimizes the least-squares error. The polynomial coefficients for the tracked velocity profile in Figure 4.9 were determine and applied to a second order polynomial equation as shown below:

$$y(t) = 141t^2 - 1149t + 3696. \tag{4.1}$$

As a comparison this polynomial was plotted and compared with the measured data as shown in Figure 4.10.

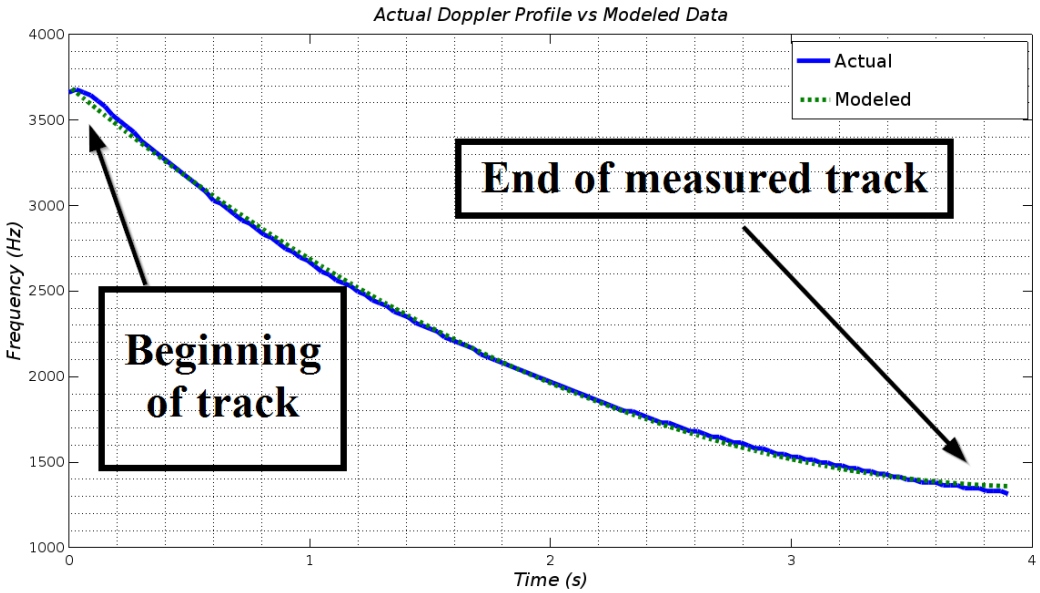


Figure 4.10: A polynomial fit on the velocity profile of the 6 Iron golf shot.

The arithmetic mean μ_{error} and standard deviation σ_{error} was taken to determine the error between the points of the two plots using the following equations [56]:

$$\mu_{error} = \frac{1}{n}\sum_{i=1}^{n} a_i, \tag{4.2}$$

$$\sigma_{error} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}. \tag{4.3}$$

These were calculated to be the following:

$$\mu_{error} = -0.48 \text{ Hz},$$

$$\sigma_{error} = 19.78 \text{ Hz}.$$

The next type of golf shot that was modeled was a Driver. The raw Doppler data and spectrogram plot in Figure 4.11 and in Figure 4.12 are shown.

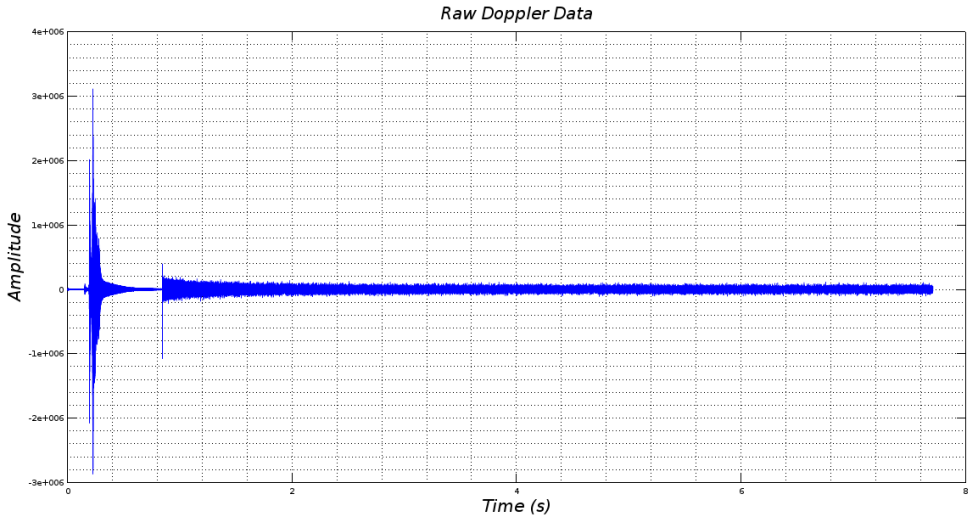


Figure 4.11: Raw Doppler data of a Driver golf shot.

Its second order polynomial equation was determined to be:

$$y(t) = 133t^2 - 1131t + 4724. \tag{4.4}$$

Furthermore, the polynomial fit was plotted and this was compared to the measured data as shown in Figure 4.13.

Then the arithmetic mean and standard deviation was taken to determine the error between these two plots:

$$\mu_{error} = 1.95 \text{ Hz},$$

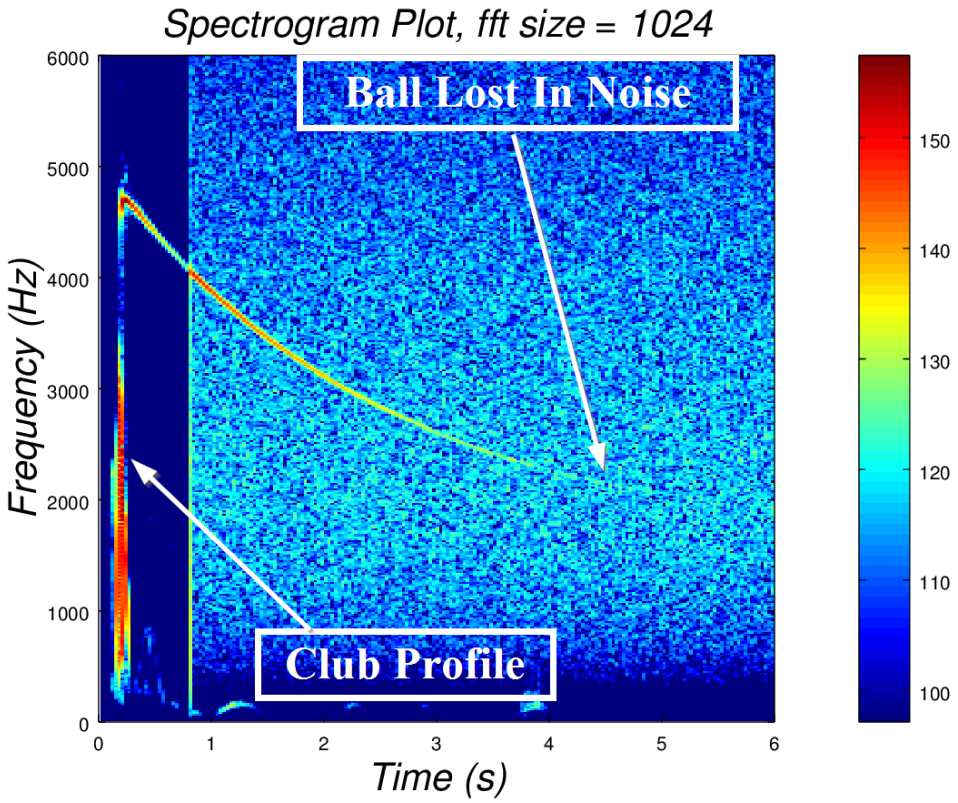


Figure 4.12: A spectrogram plot of the raw Doppler data of a Driver golf shot.

$$\sigma_{error} = 18.28 \text{ Hz}.$$

The last type of golf shot that was modeled was a Pitching Wedge. The raw Doppler and spectrogram plot is shown in Figure 4.14 and Figure 4.15 respectively.

Below is its second order polynomial fit:

$$y(t) = 117t^2 - 829t + 2665. \tag{4.5}$$

As a final comparison, the polynomial fit was plotted and compared with the measured data as shown in Figure 4.16.

Then the arithmetic mean and standard deviation was taken to determine the

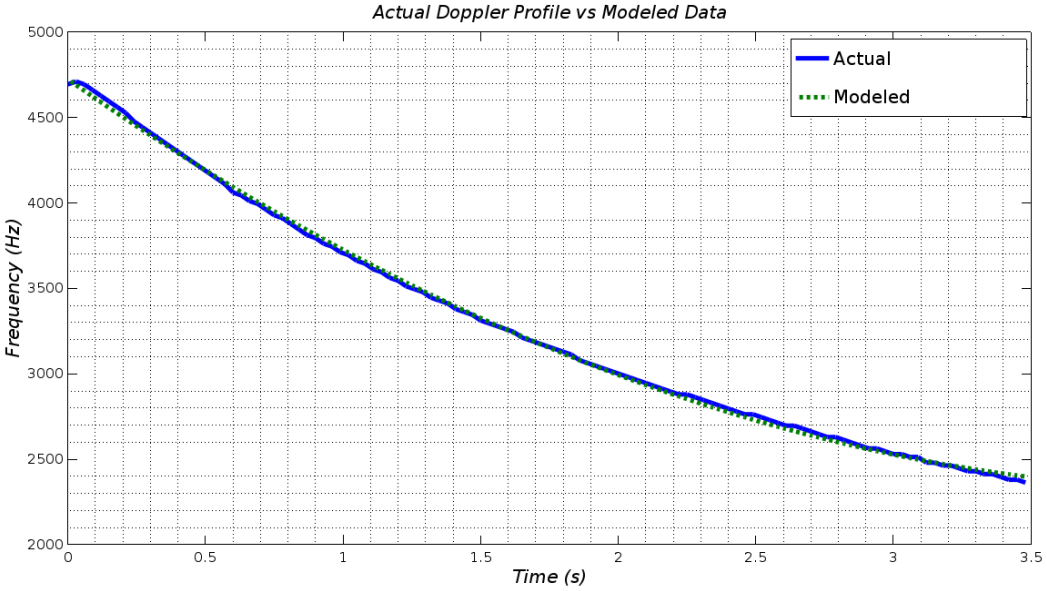


Figure 4.13: A polynomial fit of the spectrogram plot of the raw Doppler data of a Driver golf shot.

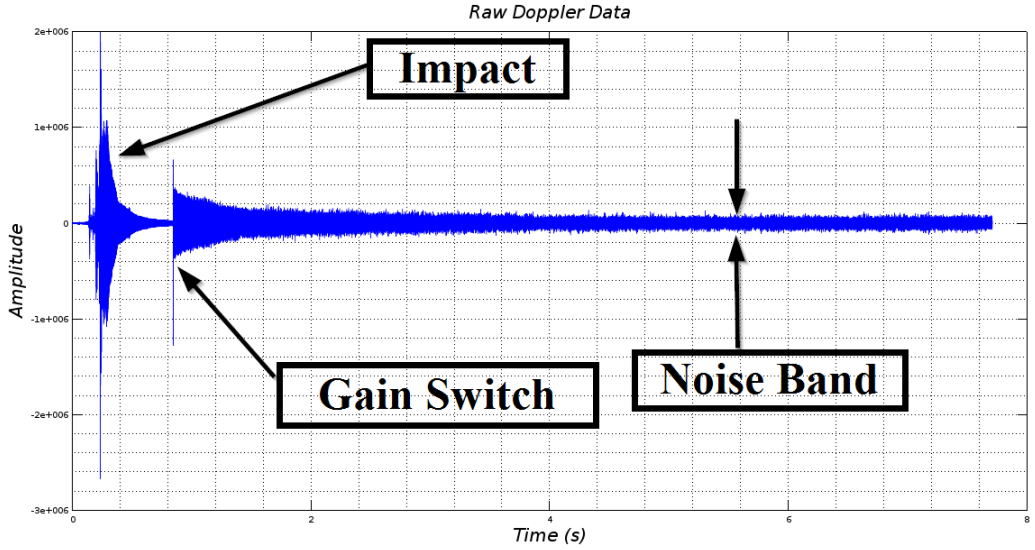


Figure 4.14: The raw Doppler data of a Pitching Wedge golf shot.

error between these two plots:

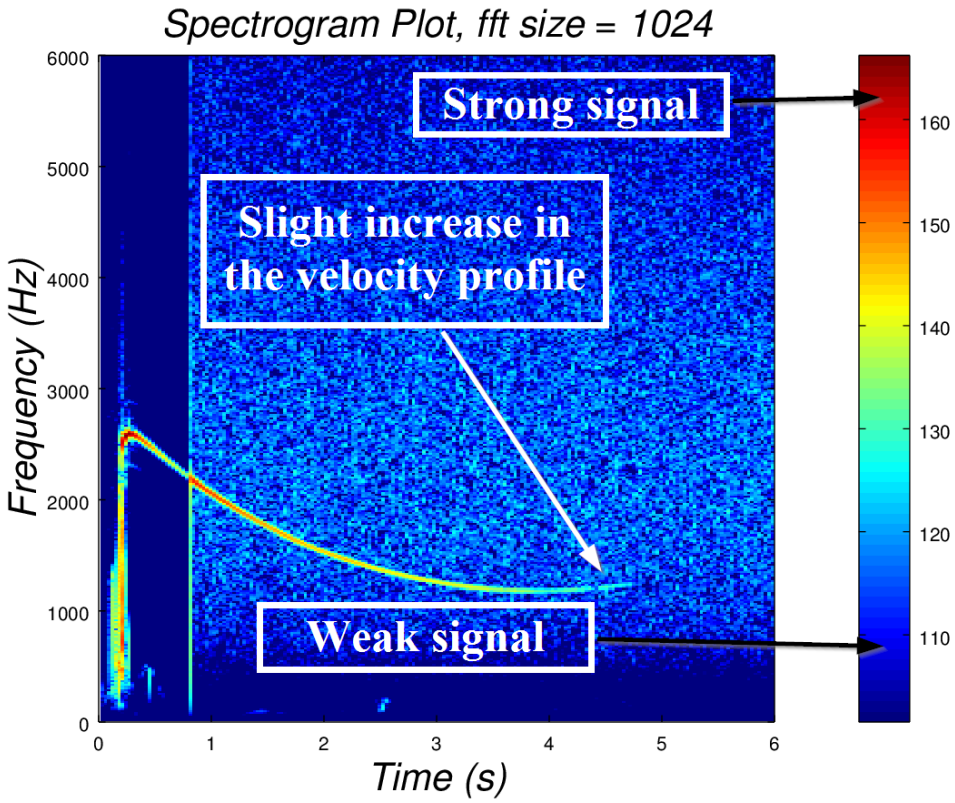


Figure 4.15: The spectrogram of the raw Doppler data of a Pitching Wedge golf shot.

$$\mu_{error} = 0.47 \text{ Hz},$$

$$\sigma_{error} = 17.82 \text{ Hz}.$$

These three equations that were derived from the spectrogram plots became the flight models. These will be used to simulate the velocity profiles. Table 4.2 below shows a summary of the modeled results in Hz.

Table 4.2: Modeled golf shot comparison versus real data in Hz

Golf Shot	μ_{error} **(Hz)**	σ_{error} **(Hz)**
Driver	1.95	18.28
6 Iron	-0.48	19.78
Pitching Wedge	0.47	17.82

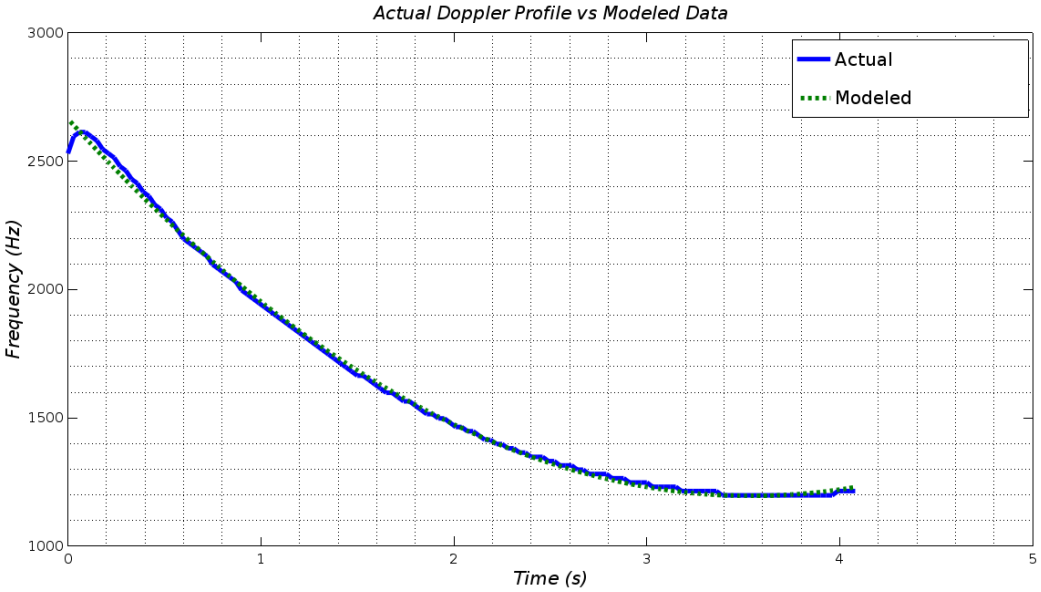


Figure 4.16: The polynomial fit of the spectrogram of the raw Doppler data of a Pitching Wedge golf shot.

4.4.3 Amplitude Profile

The literature review discussed the signal power relative to its range of a target represented by the radar range equation. Besides the velocity profile that can be deduced from the spectrogram, the decreasing amplitude profile can be determined as well. The intensity or signal strength of the velocity profile is represented by the colour bar. The frequency with the highest amplitude is represented by the colour red and lowest with the colour blue, as shown in Figure 4.15. Although this was not a design requirement for simulating the golf ball trajectory, it is still useful to include. To guarantee a ball only amplitude decay, the region from the impact point until the gain switch was ignored as this region includes club and player movement. In Figure 4.17 the derived amplitude profiles from after the gain switch was shown until the end of the tracks.

It can be observed that a similar decay profile is present in each plot. Furthermore, the Pitching Wedge appears to have a larger initial amplitude compared to the Driver and 6 Iron. This is to be expected as the Pitching Wedge has a

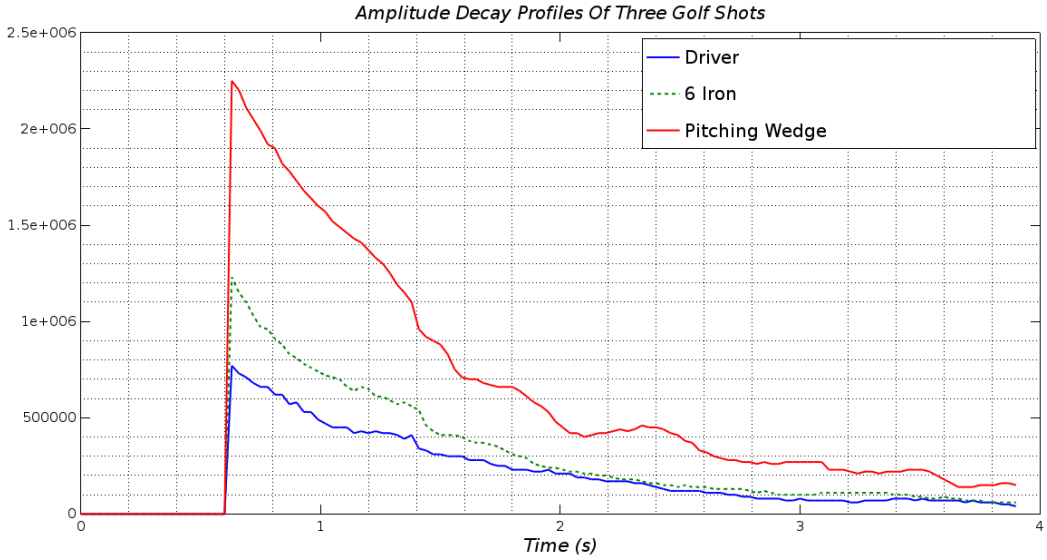


Figure 4.17: The amplitude profile of the three golf shots.

lower radial velocity at the gain switch instant compared to the other shots. This means that the Pitching Wedge shot is closer to the radar at that time instant. From these plots an exponential decay was fitted to each of these profiles. These amplitude profiles will be used to simulate each of the different type of golf shots. This can be seen in Figure 4.18.

The exponential equation has a variable called the retardation factor "R_n" which represents the attenuation of the signal. This can be seen in the equation below:

$$A(t) = A_0 e^{-R_n t}. \tag{4.6}$$

Below is a table of retardation values that was determined for each of the amplitude profiles:

Table 4.3: Amplitude Retardation Values

Shot	**Retardation**
Driver	0.0269
6 Iron	0.0251
Pitching Wedge	0.0275

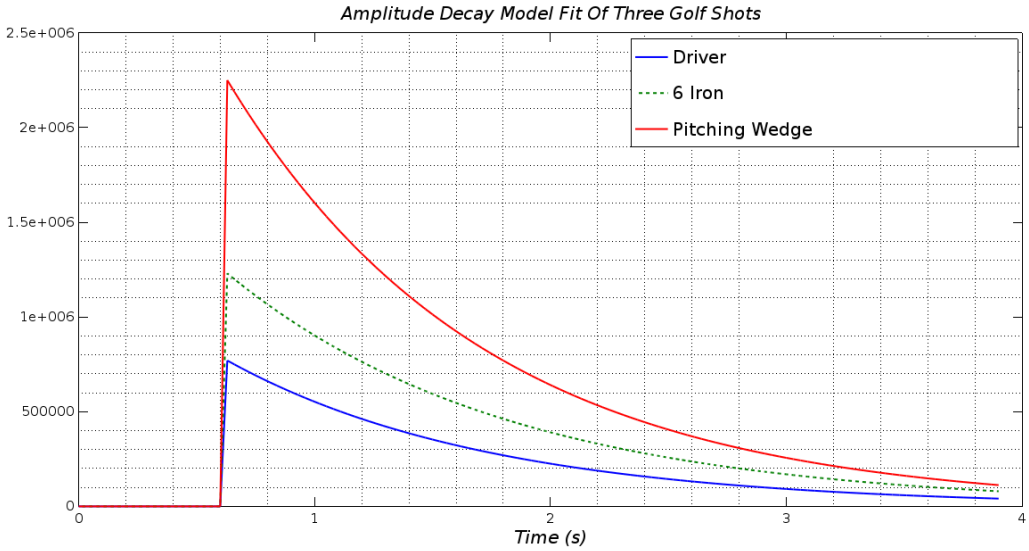


Figure 4.18: The exponential fit on the amplitude profile of the three golf shots.

It shall be seen later how amplitude modulation was implemented on the I and Q signals.

This section aimed to derive velocity profiles from actual Doppler data so that they can be used to simulate them on a Doppler simulator. In addition, a simple amplitude model was derived as well. The next section will discuss how the Due Doppler simulator will operate and the design behind the DDS method to simulate the flight models.

4.5 Firmware Design

The main function of the firmware is to operate the Due Doppler simulator in such a way that it can simulate the flight models by using the DDS method. This section will cover the firmware operation, serial communication, the DDS method, and waveform modulation.

4.5.1 Firmware Operation

The operation of the firmware for the Arduino Due is the core part of the device as it controls and operates all the components. A flow diagram of the firmware can be visualized in Figure 4.19.

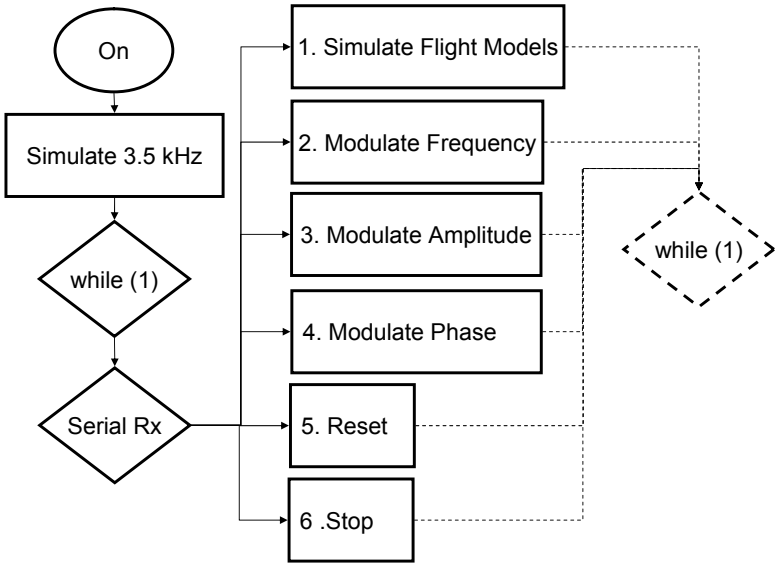


Figure 4.19: The flow diagram for the firmware for the Due Doppler simulator.

The flow diagram shows that when the Due Doppler simulator is in its "On" state, then by default it would simulate 3.5 kHz at a constant amplitude. Thereafter, it would be in the system while loop waiting for a serial message to change its mode. There are six different modes that the Due Doppler simulator can be in. The first mode is the "Simulate Flight Models" mode. This mode simulates the three different flight models that were discussed in the previous section. The next three modes all relate to when the device is in its default simulate mode. The first refers to a "Modulate Frequency" mode which allows the user to either increase or decrease the frequency by 50 Hz. The second is a "Modulate

Amplitude" mode that allows the user to increase or decrease the amplitude of the waveforms by 0.1 V. This mode also has a differential feature that allows the user to increase or decrease the amplitude of the I waveform relative to the Q waveform. The third is a "Modulate Phase" mode which is a feature that allows the user to change the phase between the I and Q waveforms by 0.35°. Then there is a "Reset" mode that resets all the modes to the default state. The final mode, "Stop", halts all simulations.

4.5.2 Serial Communication

The AD9 Doppler simulator used serial communication via an RS232 to USB converter to program and communicate with the device. A USB communication link was specified since the Due Doppler simulator uses a USB serial communication link at 9600 bps to program and communicate with the microcontroller.

The other design requirement stated that there should be a wireless communication link with the device. This would enable easier diagnostic testing of the device if any modes need to be changed. Two options are readily available for the Arduino Due, that of a Wifi or Bluetooth module. A Bluetooth module was chosen as the range of operation for the Doppler simulator is around 5 m.

4.5.3 DDS Method Design

This section will discuss the design behind the DDS method. The Arduino Due allows independent control over each of the I and Q signals. This allows for frequency, amplitude, and phase modulation and furthermore for golf ball velocity profiles to be simulated.

The literature review mentioned two parts to the DDS method. The first is the NCO and the second is the DAC. The NCO is the process in which the waveforms are generated digitally using a LUT, PA, and PI. The DAC is used to convert these digtal waveforms to analog signals. The general layout of the NCO is shown in Figure 4.20.

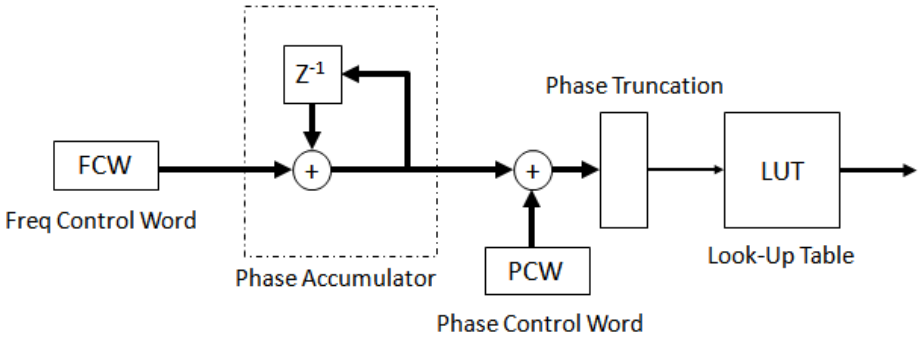


Figure 4.20: The flow diagram for DDS method [57].

Look-Up Table

The first part of the DDS method that needs to be designed is the LUT. The LUT is a table of amplitude values representing a waveform. Three aspects of the table need to be designed. These are the type of waveform, the table length (l_{LUT}), and its width. The type of waveform that was chosen was a sine waveform as it needs to fit the same structure as the RF carrier. As mentioned in the literature review, the length of the table determines the phase resolution of the generated waveforms. The chosen length was 10 bits. This allows for the following phase resolution:

$$\phi_{res} = \frac{360^\circ}{l_{\text{LUT}}} = \frac{360^\circ}{1024} = 0.35^\circ. \tag{4.7}$$

This results in a 0.35° phase modulation resolution. Then the width of the table (w_{LUT}) determines the amplitude resolution. From the datasheet of the SAM3X8E microcontroller, it showed that it has 12-bit DACs at 2.2 V each. By choosing the width to be 12 bits, the amplitude resolution can be determined through:

$$A_{res} = \frac{V_{\text{DAC}}}{w_{\text{LUT}}} = \frac{2.2}{4096} = 0.532 \text{ mV}. \tag{4.8}$$

The larger the LUT, the finer phase and amplitude resolution. However, the DAC physical characteristics limits the size of the LUT.

Phase Accumulator

The phase accumulator consists of an n-bit adder and tuning register. With each new clock-cycle a new n-bit output is generated. Each output consists of the previous output added with the PI. Over a successive amount of clock-cycles the output forms a staircase profile. This is an accumulating value that is associated with a gradient and cutoff point. The step size for the PA is proportional to the sampling rate. The SAM3X8E uses an 84 MHz clock, and allows timer prescaler values of 2, 8, 32, and 128. The AD9 Doppler simulator used a sampling rate of 20 MHz. According to the spreadsheet, the Arduino Due DAC registers uses the master clock divided by two to perform conversions which results in $\frac{84MHz}{2} = 42$ MHz. The maximum conversion rate takes 25 clocks cycles to provide an analog result so $\frac{42MHz}{25} = 1.68$ MHz or 1.68 MSps - although the datasheet stated 2 MHz. However, using the maximum sampling rate is overcompensating for the frequency range that is required. Since the largest measured frequency is less than 6 kHz, a sampling rate at 12 kHz would satisfy the Nyquist rate. However, the minimum sampling frequency of the DAC is 40 kHz. So it was decided to take a slightly larger sampling rate of 44 kHz as it satisfies the minimum DAC sampling rate with some margin. The higher sampling rate would also remove unwanted aliasing frequency components further away from the band of interest.

The next section shows how the sampling frequency was implemented using the microcontroller's timers. The next parameter that needs to be chosen is the bit resolution of the phase accumulator. The larger the bit resolution the finer the frequency resolution. To achieve a finer frequency resolution with embedded processors, fixed point arithmetic is used. This is a simple way of representing the same precision as floating point numbers by using the overflow of bits. The PA was chosen to have a 32-bit resolution. As noted in the literature review, the value of the PA references to an index value in the LUT. Since the length of the LUT is 10 bits long, there appears to be a mismatch between the number of bits used in the PA and the LUT. However, as fixed point arithmetic caters for finite increments by overflowing into success bits, the lower 22 bits is truncated from

the PA output and the upper 10 bits is used as the index for the LUT.

Phase Incrementor

The PI is one of the inputs to the PA. It refers to the tuning frequency or tuning register that determines the slope of the PA. The PI also needs to be scaled to the same bit size as the PA. This is done through the following equation which was defined in the literature review:

$$PI = f_t \frac{2^N}{f_s} \tag{4.9}$$

Furthermore the frequency resolution (f_{res}) can be determined by using the following equation:

$$f_{res} = \frac{f_s}{2^n} = \frac{44 \text{ kHz}}{2^{32}} = 0.0102 \text{ mHz}. \tag{4.10}$$

In summary the DDS parameters are the following:

Table 4.4: A parameter list for the DDS process

	Parameter	**Value**
1	LUT Length (bytes)	10
1	LUT Width (bytes)	12
2	Sample Rate (kHz)	44.0
3	Phase Accumulator (bits)	32

To validate these design criteria, a simulation was done of the DDS method in Octave.

In Figure 4.21, a correlation between the start of the PA with the start of the sine waveform can be observed. Furthermore, the end of the PA ramp and the end of the sine waveform are in line. Thereafter it resets. This, in essence, is the DDS process. The time between the beginning of the ramp relative to the end is 0.00028 s which translates to 3.5 kHz, this is the default frequency for the Doppler simulator.

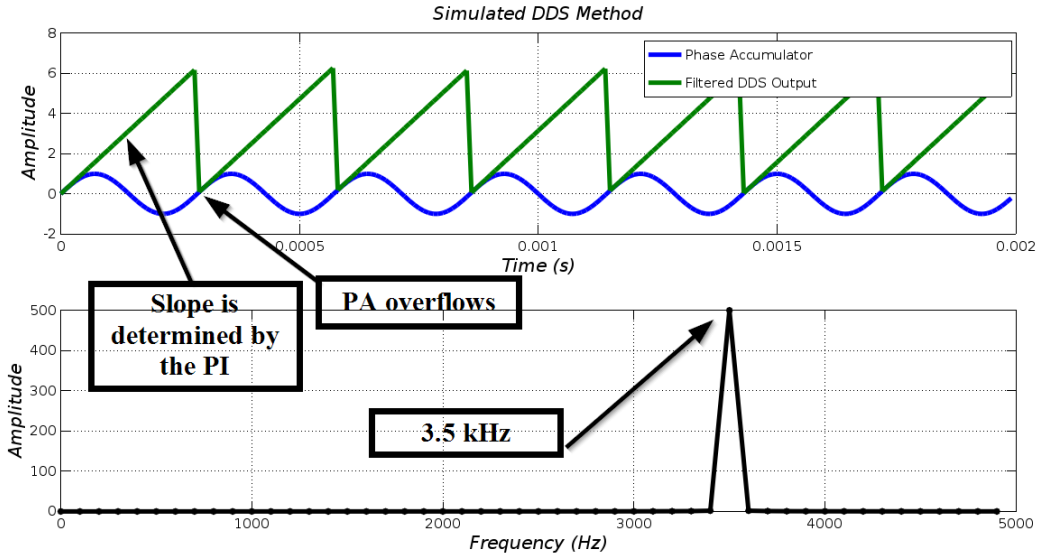


Figure 4.21: A simulation of the DDS method.

This section focused on a single waveform, this will be called the I waveform. However, the Doppler simulator requires two waveforms that are quadrature to do frequency synthesis with the RF mixer. Another waveform, called the Q waveform, needs to be generated as well. These I and Q waveforms are fed into the IF1 and IF2 pins on the RF mixer respectively. In essence, the IQ signals are exactly the same, they only have a phase offset relative to each other. The next section will show how the offset was calculated.

4.5.4 Waveform Modulation

Three types of modulations can be formed on a waveform. These are frequency, amplitude, or phase modulation. This section looks at how these three types of modulation techniques were designed to be used for the Due Doppler simulator.

To implement the flight models discussed earlier, a method of changing the frequency and amplitude at a certain rate needs to be developed. Similar to the LUT for the DDS method, separate velocity modulation LUT (VMLUT) tables for each of the flight models was designed. Since the data captured was chosen

to stop at 4.5 s, the length of the LUT tables were taken to be this value. The width of the LUT tables for frequency modulation ranged from 1000 to 5000 Hz and for amplitude modulation it was taken proportional to the maximum voltage range of the DAC which is 2.2 V. As noted before, the DDS LUT had a sampling rate (f_s) of 44 kHz. Similarly, the modulation LUTs needed to be sampled at a much slower sampling frequency. This was determined from the spectrogram which used 17045 as its sampling frequency (f_{spec}) which came from the Doppler radar. To determine the length of the VMLUT (l_{VMLUT}), the VMLUT delta (δt_{VMLUT}) is first determined as follows,

$$\delta t_{\text{VMLUT}} = \frac{\text{window - overlap}}{f_{spec}} = \frac{1024 - 512}{17045} = 0.03 \text{ s}. \tag{4.11}$$

Consequently, for a 4.5 s track length, the LUT length becomes

$$l_{\text{VMLUT}} = \frac{4.5}{0.03} = 150. \tag{4.12}$$

The final type of modulation that needed to be designed was phase modulation. The only design requirement was that these IQ waveforms needed to be 90° out of phase to ensure a SSB from the RF mixer. As mentioned earlier, the length of the LUT is a representation of the phase resolution. Furthermore, to achieve the quadrature waveforms the Q waveform needs to be lagging by 90° to simulate targets moving away. Since 90° refers to a quarter of a full waveform of 360°, the start of the Q waveform would need to be offset by a quarter of the length of the LUT. The configuration of the Doppler radar uses a lagging Q to represent objects moving away. The delta offset for the Q waveform (δ_{Q}) is calculated as

$$\delta_{\text{Q}} = \frac{l_{\text{LUT}}}{4} = \frac{1024}{4} = 256. \tag{4.13}$$

4.6 Electronic Hardware Design

This section will discuss the electronic hardware that is required to perform the simulation set out by the firmware as well as the peripheral components. The

Arduino Due will be discussed in terms of its functionality and operation. This section will also discuss the design of the passive filters, serial communication, power consumption, and the general layout of all these components. Appendix A shows some snippets of the datasheets used to design the components in this section.

4.6.1 Arduino Due Overview

The Arduino Due is a development board that uses a 32-bit Arm core microntroller. It has the following specifications:

- Operating Voltage: 3.3 V
- Input Voltage: 7-12 V
- Digital I/O Pins: 54 (12 are PWM)
- ADC Pins: 12
- DAC Pins: 2
- Total DC Output Current: 130 mA
- DC Current for 3.3V Pin: 800 mA
- DC Current for 5V Pin: 800 mA
- Flash Memory: 512 KiB
- SRAM: 96 KiB
- Clock Speed: 84 MHz
- Length: 101.52 mm
- Width: 53.3 mm
- Height: 15 mm

- Weight 36 g

A top view image of the Adruino Due is shown in Figure 4.22.

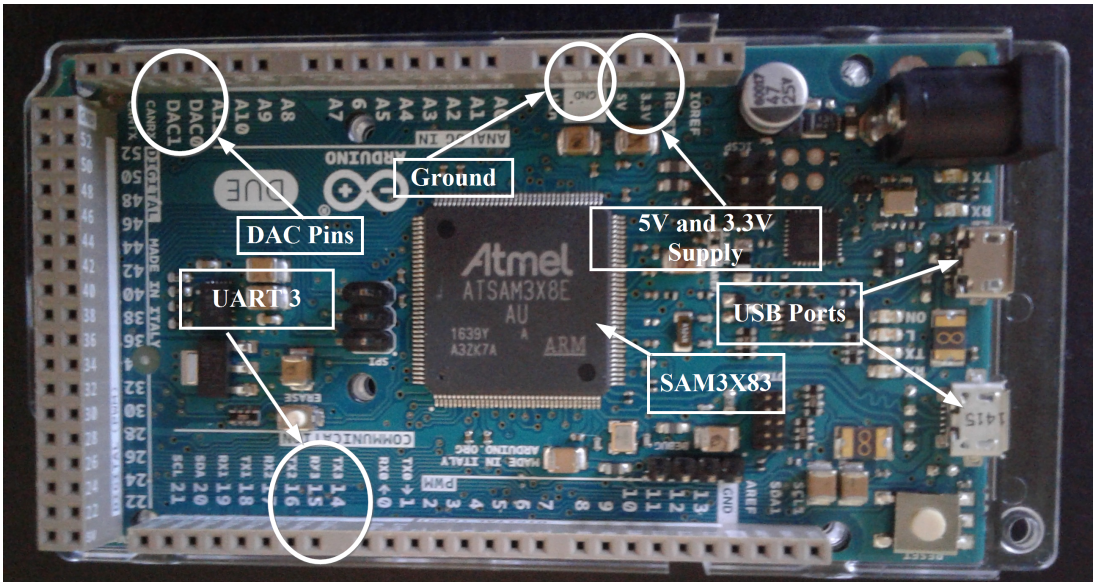


Figure 4.22: A top view perspective of the Arduino Due

The main feature that the Arduino Due possess over the other Arduino development boards, is that it has two dedicated 12-bit DACs. Its 84 MHz Arm processor is also unique as it allows for high speed processing and a high sampling rate for the DDS method and DAC.

4.6.2 Passive Filter Design

It is known that the DDS method uses the DAC to convert the digital waveforms to analog signals. However, this process results in two side effects. This first is the quantization of the analog waveform and the second is a DC offset.

To filter the quantization, a first order low-pass filter needed to be designed. This was added in line with the DAC pins. The main criteria for this filter is the cutoff frequency. Since the largest frequency is under 5 kHz, a cutoff frequency

of 15 kHz was chosen which is represented by the following equation:

$$f_c = \frac{1}{2\pi RC} \tag{4.14}$$

The appropriate resistor and capacitor values were chosen as: R = 100 Ω and C = 0.1 μF. In order to remove the DC offset or bias from the DAC signals, a first order high-pass filter would need to be designed and then added in line with the ouput of the LPFs. As with the LPF, the main design criterion is the cutoff frequency which was designed to cater for the lowest frequency. From the previous section, the lowest frequency was under 1.2 kHz. Subsequently, a cutoff frequency of 500 Hz was chosen. The appropriate resistor and capacitor values were chosen to fit this cutoff frequency and the impedance of the mixer. These were chosen to be R = 330 Ω and C = 1 μF. Another important consideration is the input pins of the RF mixer. According to its datasheet, these pins are DC coupled. Furthermore, the datasheet states that a series blocking capacitor should be used to pass the necessary IF frequency. This was fulfilled by the capacitor in the highpass filter.

4.6.3 Serial Communication

The AD9 Doppler simulator used serial communication via a RS232 to USB converter. Unlike the Arduino Uno that has a single UART, the Arduino Due has 4 UARTs that can be used. The default Arduino baud rate was chosen as 9600 bps and initialized in the firmware. As stated in the design requirements, the upgraded Doppler simulator needs to be operated wirelessly. Since the operating range is under 10m, using Bluetooth would be an ideal solution. A common Bluetooth module that is used for Arduino boards is the HC-06 module and is shown in Figure 4.23.

These modules also use serial communication to communicate with an Arduino board. Since the Arduino Due has multiple UARTs, this allows for the functionality of simultaneous communication between USB and Bluetooth interfaces. UART 3 was used for the Bluetooth module and the baud rate was chosen as

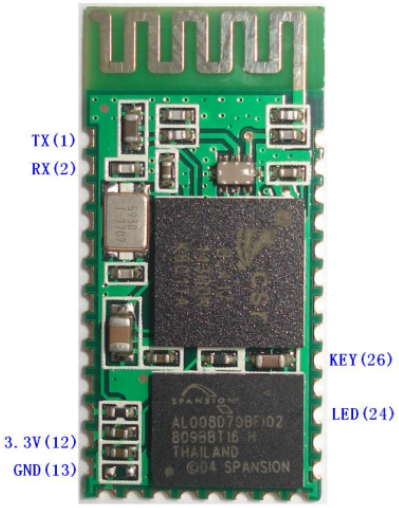


Figure 4.23: Bluetooth HC-06 Module [58]

9600 bps.

The Arduino IDE has a built-in serial monitor that allows serial communcation to take place. For the Bluetooth interface, a smartphone app called "Serial Bluetooth Terminal" was used to communicate with the Arduino [59]. A schematic diagram of these components integrated together can be seen in Figure 4.24. Further details of how this was implemented will be discussed in the next chapter.

4.6.4 Power Consumption

The Due Doppler simulator has two components that require a voltage source from the Arduino Due, namely the amplifier on the RF board and the Bluetooth module. According to the Bluetooth datasheet, it draws 8 mA at 3.3 V while communicating. Then the RF board requires 5 V for the amplifier. Furthermore, the two DACs initially supplied 2.2 V. However, this was too much for the RF mixer, so the voltage supply was adjusted to 0.4 V for the IF pins on the RF mixer. The resistor from the LPF, 100 Ω, and the load impedance from the RF mixer, 50 Ω, translates to a driving current 2.6 mA. Which is suitable as it is

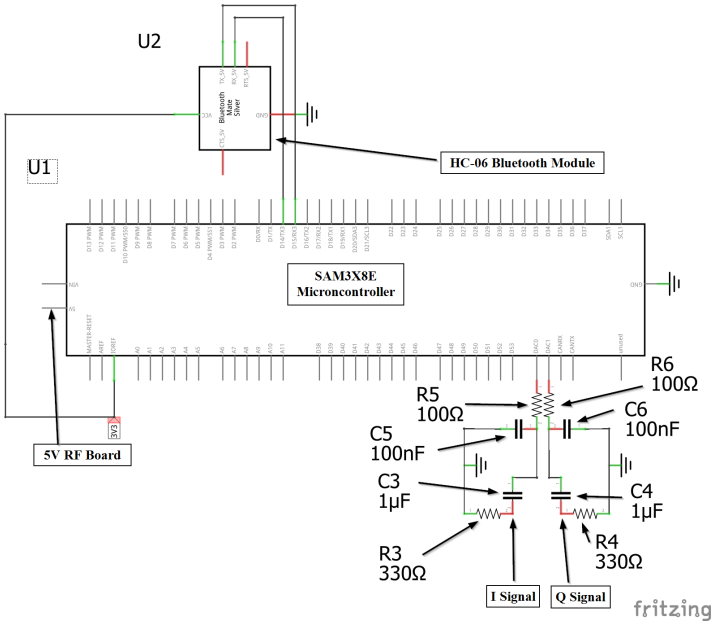


Figure 4.24: A schematic diagram for the electronic components.

below the maximum source current of 3 mA for the IF pins.

One final design requirement was to make this device portable so it would run on an external DC power supply. The voltage supply on the USB port is 5 V with the maximum current rating on the Arduino Due at 800 mA. To meet the requirements of 10 minutes usage time, a battery bank was chosen to use which had a capacity of 2000 mAh.

Figure 4.25 shows the general layout of the Arduino Due, Bluetooth module, and passive filters. A parts list for the components can be found in Appendix D.

4.7 Chapter Summary

This chapter covered the design behind the Due Doppler simulator. It first began by analyzing the operation and features of the AD9 Doppler simulator. It then discussed how the AD9 Doppler simulator would be upgraded to become the Due Doppler simulator in terms of its operation. The chapter then transitioned to

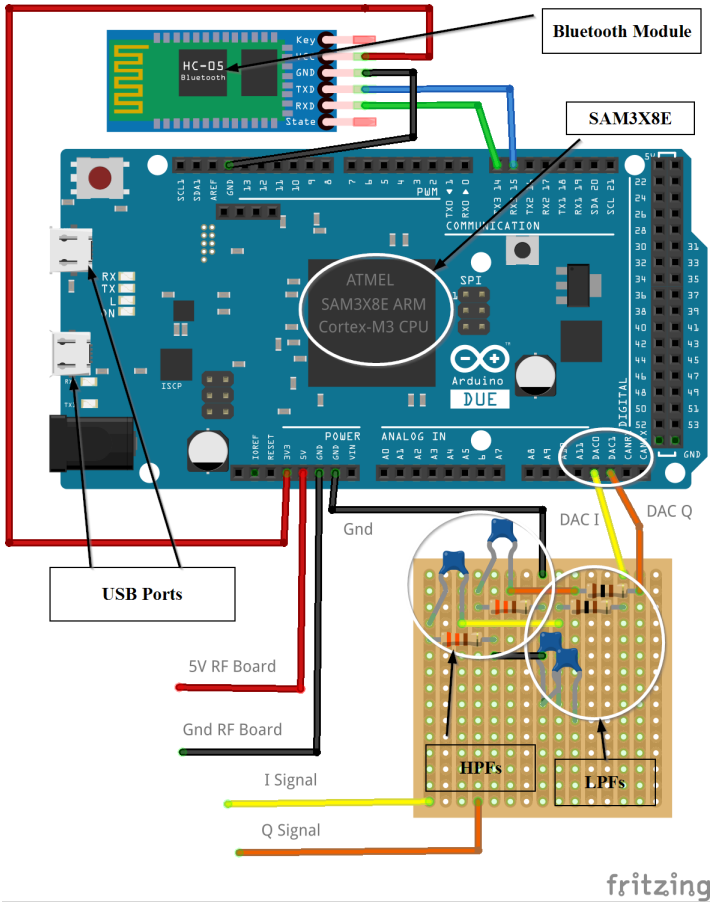


Figure 4.25: Electronic Hardware Layout

deriving flight models for three golf shots from empirical data. It then discussed the firmware design by focussing how it would meet the design requirements with a specific focus on the DDS method and waveform modulation. The final section of this chapter discussed the design of the electronic hardware.

Chapter 5

Firmware and Hardware Integration

This section will focus on implementing all the aspects discussed in chapter 4. The focus of this research is to upgrade the AD9 Doppler simulator with electronic hardware and firmware that is capable of simulating a golf ball trajectory.

The implementation was performed in two stages. The first stage was to implement the firmware designs mentioned in the previous chapter. The second was to integrate the Arduino Due with the RF board. This consisted of powering the RF board and connecting the IQ signals to the two IF inputs in the mixer.

5.1 Firmware Implementation

The main purpose of the Arduino Due board is to implement the DDS method so that it can generate IQ signals that simulate the golf model profile. The implementation of the DDS method will be covered in three sections which will discuss the Arduino IDE, the main firmware functions, and the general firmware operation.

The Arduino series of development boards uses a custom Arduino IDE that

supports C and C++. The Arduino IDE also allows for a serial interface to inspect messages on the serial ports. The contents of any Arduino program are made up of files called “Sketches”. These “Sketches” are divided into three parts, namely structures, values, and functions.

As with any piece of embedded code, it requires variable definitions that are going to be used throughout the program. The compiler will interpret these variables in various ways depending on how they are defined. The Due Doppler simulator firmware used a variety of variables for timers, the DACs, the DDS, serial communication, and modulation functions. Standard variable types were used like integers, floats and chars. Other variable types were also used like hash defines and volatile variables inside the interrupt routines.

The following list shows the layout of the firmware which comprises of 6 functions:

1. Create DDS Table
2. Create MOD Table
3. Setup
4. Loop
5. Serial Handler
6. DDS Handler
7. MOD Handler

Create DDS LUT

This function is initialized at startup in the Arduino’s “Setup” function. This function uses a for loop to populate the DDS LUT array for a sine wave. As mentioned in the design chapter, the size of the table is 12 by 10 bit. Since the DAC does not generate negative voltages the range of amplitude values was shifted between 0 to 4095.

Create MOD LUT

This function is also initialized at startup in the Arduino's "Setup" function. This function contains a for loop that populates the LUTs for the frequency and amplitude modulations that simulate the flight models discussed previously. The width of the LUTs are determined by the modeled amplitudes extracted from the frequency and amplitude plots. The length of all the LUTs were set to 4.5 s by using a 30 ms step interval determined by the spectrogram of the Doppler data.

Setup

The "Setup" function is a standard Arduino function that forms part of the main structure of any Arduino "Sketch" program. It is used to initialize variables, serial communication, functions, timers, pin modes, and other libraries. The Due Sketch used the "Setup" function to initialize the variables mentioned earlier as well as the "DDS" and "MOD LUT" functions. It also initialized the USB and Bluetooth serial communication at a baud rate of 9600, as well as the two DACs whose registers were initialized to 12 bits. The two timers were also initialized in this function. These are the timer 0 for the waveform modulation and timer 1 for the DDS method. Timer 0 is an 8-bit timer that is used to implement the standard timer functions such as "delay()" and "millis()". The Due Sketch uses the "millis()" timer function to update the interrupt service routing (ISR) to simulate the waveform modulation at an interval of 30 milliseconds. Timer 1 is a 16-bit timer whose registers are setup to generate the sampling frequency of 44 kHz that was discussed previously.

Loop

The "loop" function is similar to a "main" function in standard C IDEs. It occurs after the "Setup" function and runs in a continuous loop performing whatever instructions or functions it contains. Only two functions are called here, these

are the “MOD Handler” and the “Serial Handler”.

Serial Handler

This function handles all the serial communication that is either from the USB or Bluetooth ports. The USB serial communication was handled by UART 1 and the Bluetooth serial communication was handled by UART 3. The following modes handled by this function are the following:

1. Simulate 3.5 kHz
2. Simulate Flight Models
   (a) Driver Flight Model
   (b) 6 Iron Flight Model
   (c) Pitching Wedge Flight Model
3. Modulate Frequency
   (a) Increase by 50 Hz
   (b) Decrease by 50 Hz
4. Modulate Amplitude
   (a) Increase by 10 %
   (b) Decrease by 10 %
   (c) Differential increase by 10 %
   (d) Differential decrease by 10 %
5. Modulate Phase
   (a) Increase the phase difference by 0.35°
   (b) Decrease the phase difference by 0.35°
6. Reset

7. Stop

Once a serial command is sent via USB or Bluetooth, a debug message is returned stating the mode change. An example of these messages is shown via USB and Bluetooth in Figure 5.1 and Figure 5.2

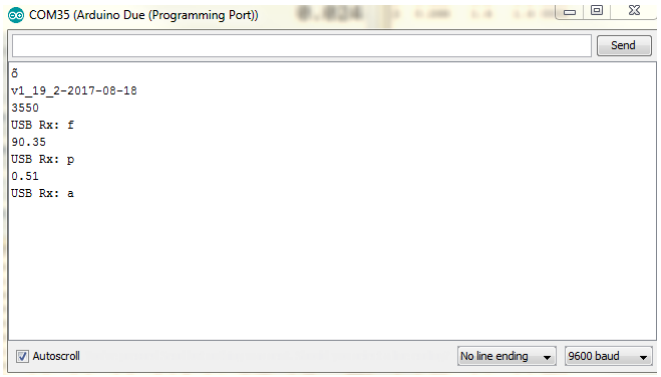


Figure 5.1: A snapshot of the serial monitor used to communicate with the Due Doppler simulator via USB

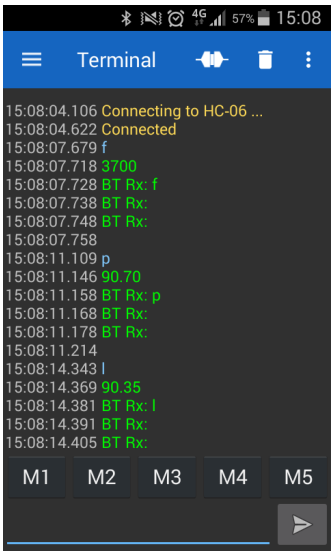


Figure 5.2: A snapshot of the app used to communicate via Bluetooth with the Due Doppler simulator

DDS Handler

The DDS handler uses the ISR timer 1 to initiate an interrupt at rate of 44 kHz. At every instant that the interrupt is called the DDS method is performed. As shown in the design phase, the DDS method is characterized by the NCO and DAC. The amplitude value is extracted from the LUT and then used by the DAC to generate an analog value. Since two DACs are used to generate the IQ signals, the Q signal is shifted by 90° by a phase offset. The next section will show how the amplitude, PI, and phase offset are the three variables that will be updated to perform the amplitude, frequency, and phase modulation.

Modulation Handler

This function uses timer 0 to access LUT tables of the flight models. Similar to how the "DDS Handler" initiates a timer, this interrupt is initiated every 30 ms as per the spectrogram step rate. At each interrupt instance, the next frequency and amplitude value is enumerated in the LUTs and used in the DDS Handler function.

5.2 Electronic Hardware Integration

The electronic hardware that was designed for the Due Doppler simulator consists of the Arduino Due and the accessory board. It was integrated into the housing by firstly taking the AD9 DDS board and phase shifter board out of the housing and replacing it with the Arduino Due and its accessory board. As per the design layout, the two DACs pins, 76 and 78, were wired to the input pins to the LPFs on the accessory board. Pin 76 generates the I signal and pin 78 generates the Q signal. Then the RF boards IQ header was connected to the output of the HPFs. The Bluetooth module was connected to the accessory board and its TX and RX pins were wired to the Arduino Due RX3 and TX3 pins respectively. Finally the accessory board was connected to its 3.3 V supply for the Bluetooth

module and the RF board was wired with its 5 V supply from the Arduino Due board. The top view of the Due Doppler simulator can be seen in Figure 5.3 as well as the accessory board in Figure 5.4.

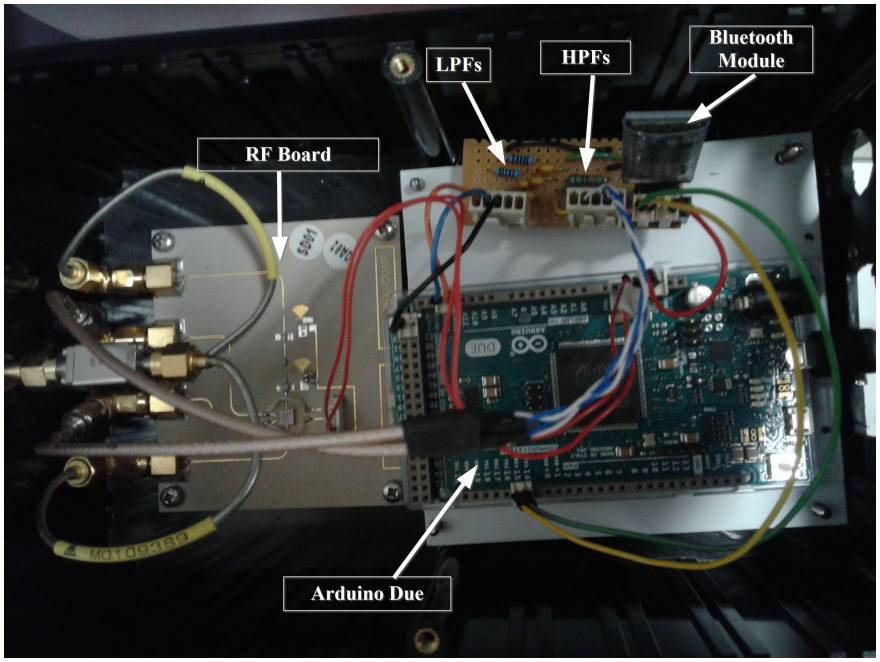


Figure 5.3: A top view perspective of the Arduino Due integrated into the housing.

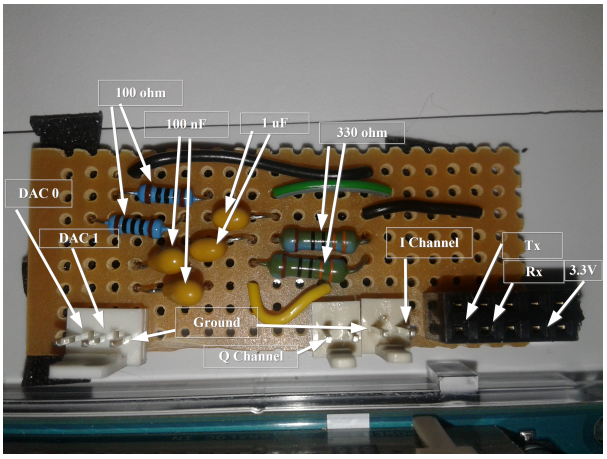


Figure 5.4: A top view of the accessory board.

5.3 Chapter Sumary

This chapter discussed how the firmware and electronic hardware was integrated into the AD9 Doppler simulator. The key features that were discussed was the firmware structure of the Due Doppler simulator and how it implemented the design features discussed in the Chapter 4. Then a brief discussion was provided on the electronic hardware which showed how the Arduino Due was placed and connected inside the housing. The accessory board and its relevant components were also shown.

Chapter 6

Evaluation of the Due Doppler Simulator

The previous chapter focused on how the electronic hardware and firmware was implemented into the Due Doppler simulator. This chapter will evaluate the Due Doppler simulator using RF test equipment and with a Doppler radar. The general functionality of the device will be tested using an oscilloscope and spectrum analyzer. Then the device will be evaluated by simulating various Doppler velocity profiles that is measured by a Doppler radar.

6.1 Evaluation under Controlled Environment

In order to test the functionality of the Due Doppler simulator, the IQ signals need to be validated to make sure they are being properly mixed and then transmitted through the antenna. Since this is a passive device, an external X-band signal needs to be generated. This was done by using a frequency oscillator and an X-band synthesizer connected to an X-band horn antenna that acted as a transmitter. Another X-band horn antenna was connected to a spectrum analyzer to act as the receiver. The layout of this can be seen in Figure 6.1. The frequency synthesizer was powered by an external power source with two voltage

inputs of 15 V and 6 V. It also had a reference signal that it got from an oscillator which was powered by 6 V. Further details on the test equipment is shown in Appendix E.

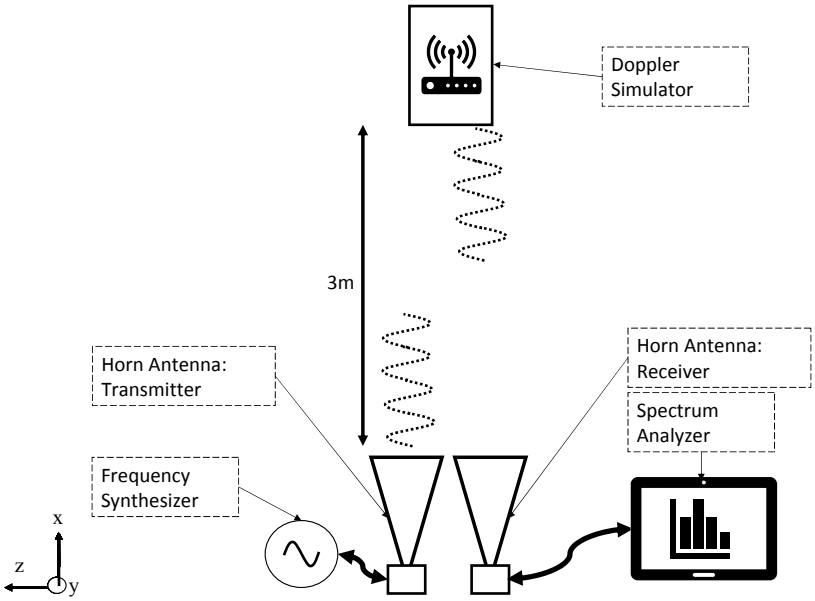


Figure 6.1: A snapshot of the spectrum analyzer of the AD9 Doppler simulator off while the test X-band hand is transmitting.

The distance from the Doppler simulator to the X-band horn antennas was 3m. It was also at the same height and in line with the horn antennas. The X-band horns were used to simulate a Doppler radar. Since angles are not being simulated, the relative horizontal or vertical position of the Doppler simulator was not important. Any offset would only effect the peak amplitude that is measured. Further details of this equipment is discussed in Appendix E. The spectrum analyzer was setup to span a spectrum of 30 kHz as a narrow bandwidth was required around the carrier frequency. During the actual testing process, the Due Doppler simulator was powered on and made to simulate a constant frequency of 3.5 kHz.

It can be observed in Figure 6.2 that a signal of 10.520 Ghz is being measured

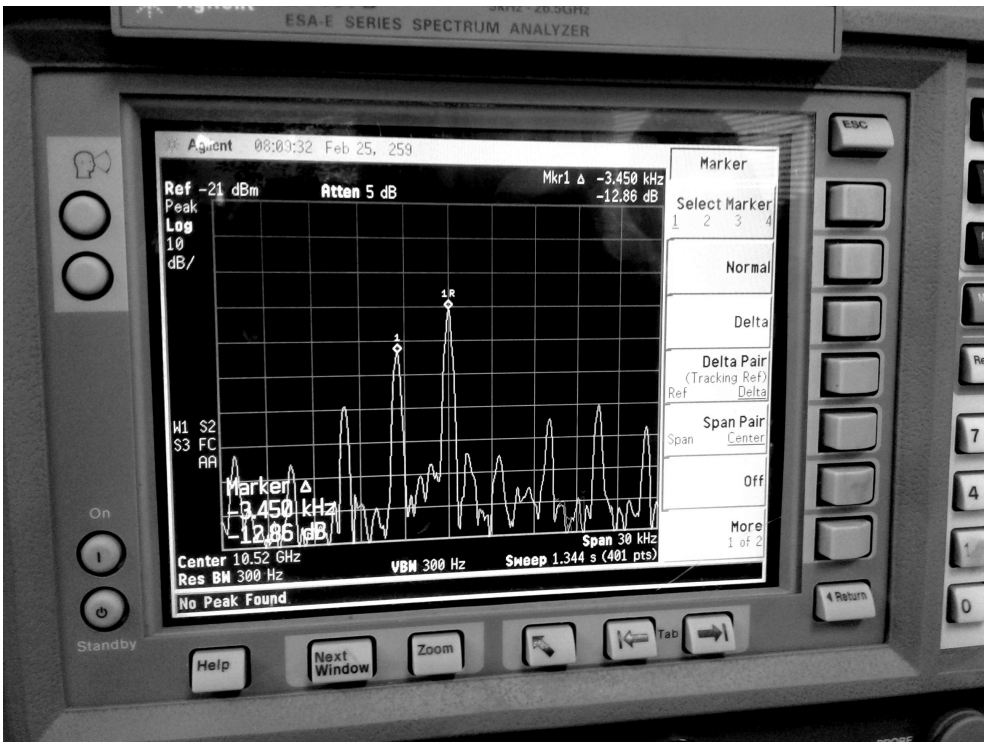


Figure 6.2: The raw Doppler data captured by the Doppler radar of 3.5 kHz simulated by the Due Doppler simulator

by the spectrum analyzer and smaller peaks around it.

It was found that the delta between the carrier and the first lower sideband is 3.45 kHz which confirms that the Due Doppler simulator was mixing the IQ signals correctly. However, there seemed to be an offset of 50 Hz. This offset can be explained by the course resolution of the spectrum analyzer which is approximately 75 Hz. It can also be observed that other peaks are present in the spectrum. This is to be expected since this is an effect of intermodulation caused by RF mixing. Another configuration of the setup showed how the USB was suppressed, as shown in Figure 6.3.

It can be observed that the USB was suppressed by -39.66 dB relative to the first LSB. The USB suppression was achieved by tuning the phase offset. In order to achieve the design requirement of a 40 dB sideband suppression, the phase offset needed to be adjusted to 77°. The final test was to evaluate the Due Doppler

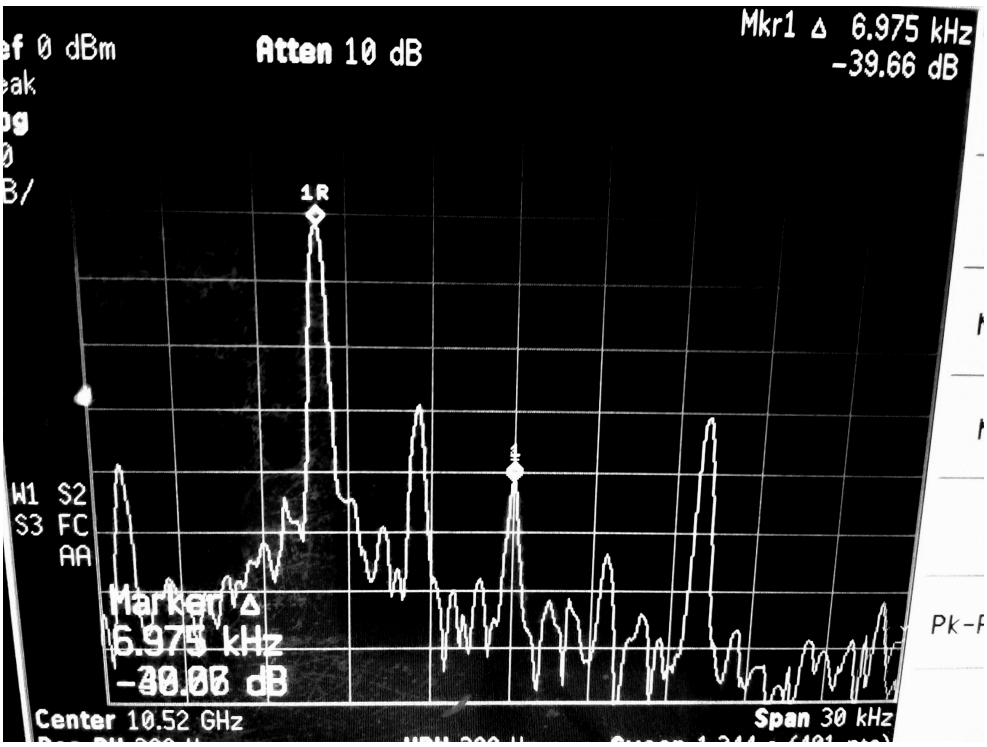


Figure 6.3: USB suppression relative to the LSB.

simulator with a Doppler radar. This will be covered in the next section.

6.2 Evaluation with a Doppler Radar

The previous section showed how the Due Doppler simulator successfully generated IQ signals that were detected as LSB peaks on the spectrum analyzer. The next stage of the evaluation is to test the device with a FlightScope Doppler radar. The Doppler radar was setup in a golf hitting cage in its normal position, that is 3 m behind the Due Doppler simulator as shown in Figure 6.4.

The Doppler radar was setup to be in an arm state which allowed it to continuously transmit 10.5 GHz. Then the Doppler simulator was setup to simulate the constant velocity profile of 3.5 kHz or 50 m/s. Then after 5 seconds, the Doppler radar was disarmed. The radar itself has a circular buffer that records

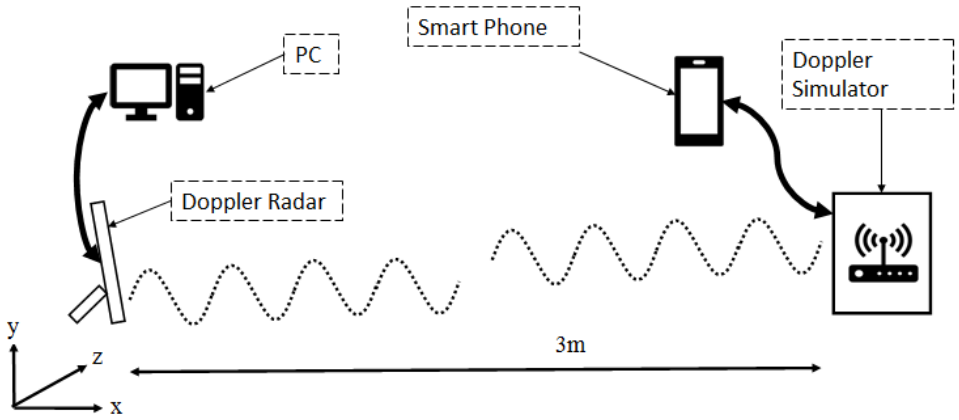


Figure 6.4: The Doppler radar setup relative to the Due Doppler simulator.

the Doppler data. This data was then downloaded from one of its receiver channels. This data was then analyzed in Octave. The raw Doppler data can be observed in Figure 6.5 and the spectrogram is shown in Figure 6.6.

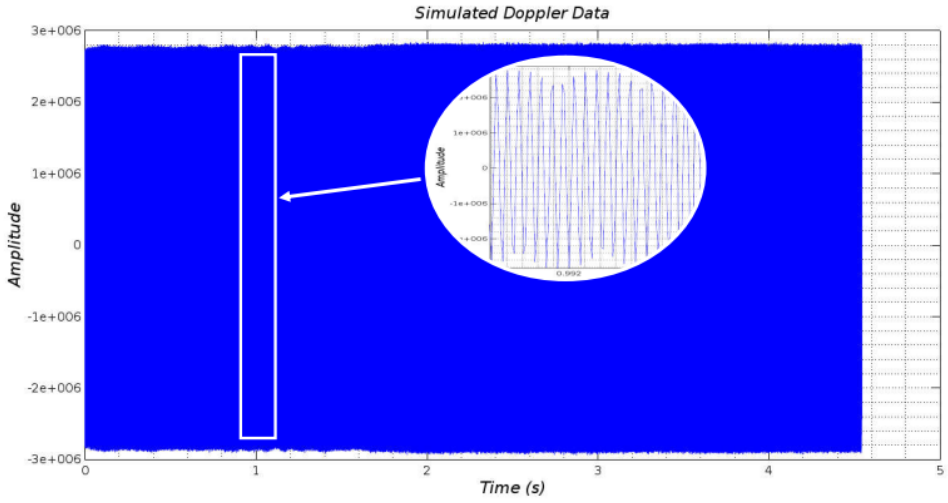


Figure 6.5: The raw Doppler data captured by the Doppler radar of 3.5 kHz simulated by the Due Doppler simulator

It can be observed that the Doppler radar captured the 3.5 kHz signal simulated by the Doppler simulator at a constant amplitude. The next evaluation was to perform the simulation of the golf ball velocity profiles. The Doppler radar was again put in its arm state and then the mode of the Due Doppler simulator was

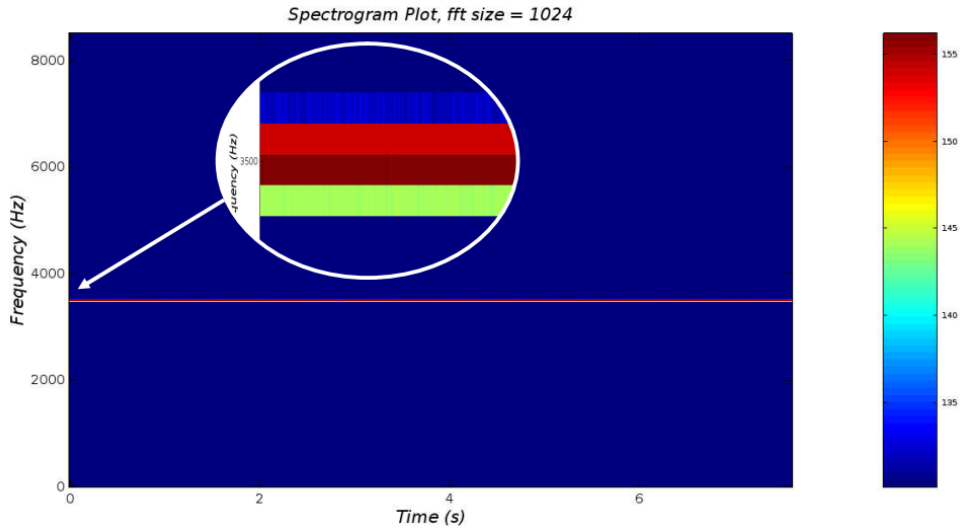


Figure 6.6: The spectrogram of the raw Doppler data of the simulated 3.5 kHz

changed to simulate the velocity profile of a 6 Iron golf shot. After 5 s the Doppler radar was disarmed. Then the data was downloaded and observed in Octave. A decaying amplitude profile can be observed in the raw Doppler data of Figure 6.7. Furthermore, the spectrogram plot shown in Figure 6.8 shows the velocity profile of the 6 Iron golf shot. It can also be observed that there is a second trajectory that is above the main velocity profile. This is the intermodulation product that is inherent with multiplying signals in a RF mixer. This harmonic is clearly a lot weaker than the main trajectory as it only appears for 800 ms on the spectrogram.

Then a comparison was done regarding how close this velocity profile matched the real shot taken by the Doppler radar. This can be see in Figure 6.9. It can be observed that there is a slight deviation from the simulated velocity.

From Table 6.1, it can be observed that the μ_{error} has a large difference of -30.47 Hz compared to the modeled μ_{error}. While the σ_{error} is more comparable. The reasons for this shall be discussed in the next section.

Following on from the 6 Iron golf shot, the other two velocity profiles were simulated. The same procedure was used as before to simulate the velocity profile. The Doppler simulator was setup to transmit a Driver velocity profile. The

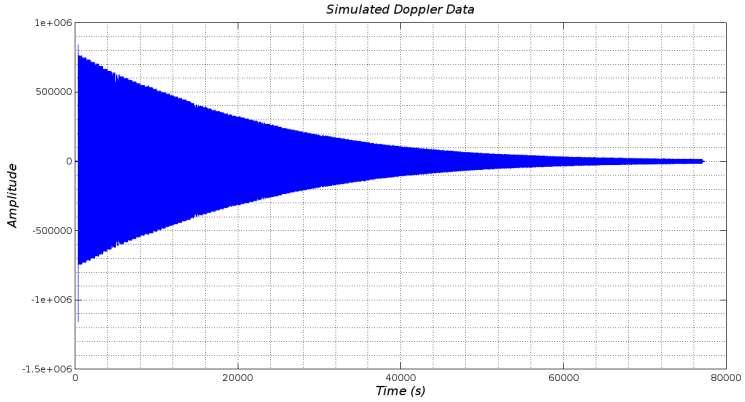


Figure 6.7: 6 Iron Raw Doppler

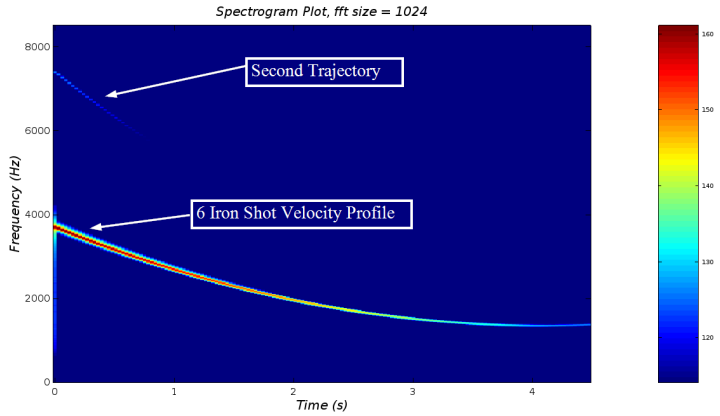


Figure 6.8: 6 Iron Spectrogram

Doppler radar was armed and then disarmed after 5 s. The Doppler data was download and the resulting data is shown in Figure 6.10 and Figure 6.11.

Figure 6.12 shows a comparison between the actual, modeled, and simulated velocity profiles.

It can be observed in the figures that the Driver has the highest launch speed and a lower deceleration rate compared to the 6 Iron. Another polynomial is fitted to this spectrogram to compare it to the modeled and real profile captured by the Doppler radar. It can be observed in Table 6.2 that the μ_{error} has a large difference of -26.03 Hz compared to the modeled μ_{error}. While the σ_{error} is comparable. The reasons for this shall be discussed in the next section.

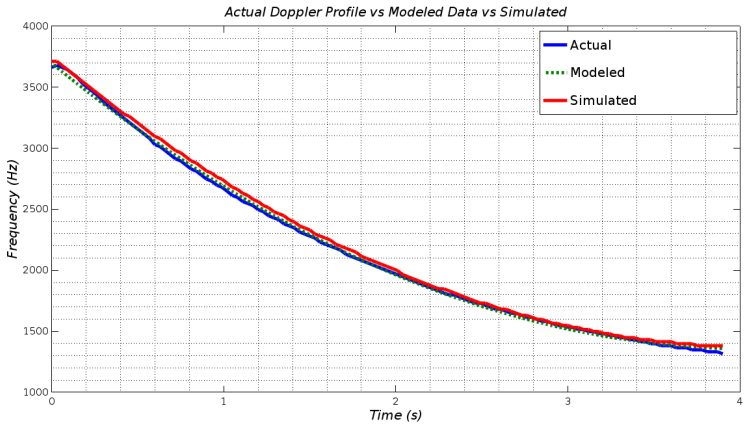


Figure 6.9: The plot of the simulated 6 Iron shot versus the modeled and real data

Table 6.1: A comparison between the modeled and simulated plots relative to the real plot for the 6 Iron golf shot

Plot	μ_{error} **(Hz)**	σ_{error} **(Hz)**
Modeled	-0.48	33.01
Simulated	-30.47	21.92

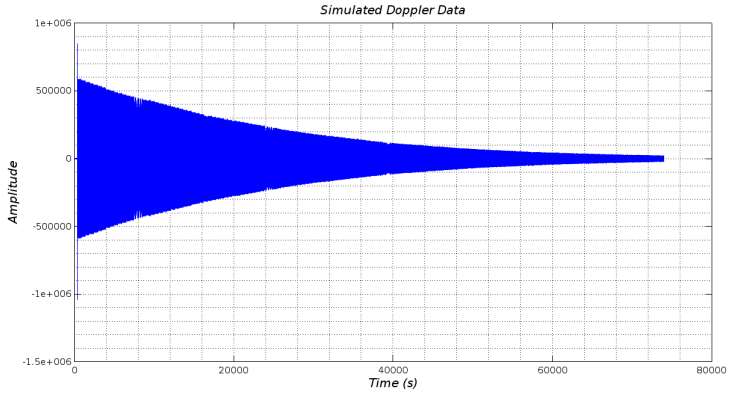


Figure 6.10: Driver Raw Doppler

The last velocity profile to be simulated is the Pitching Wedge. This type of shot has the shortest distance, lowest speed, but the highest launch angle. The same procedure to capture the simulated data with the Doppler radar is followed. The only difference is that the Due Doppler simulator is setup to simulate a Pitching

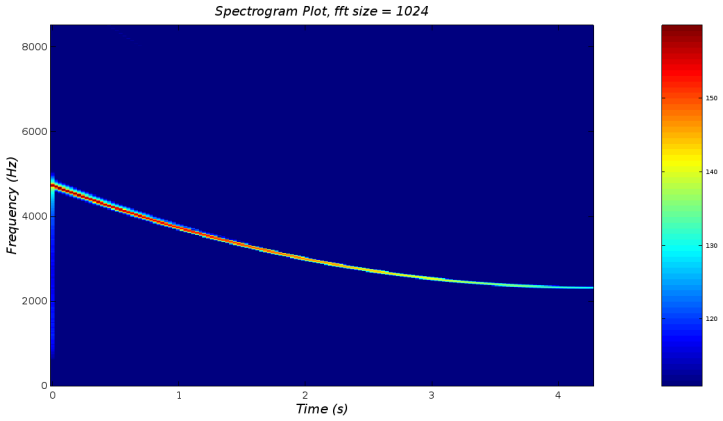


Figure 6.11: Driver Spectrogram

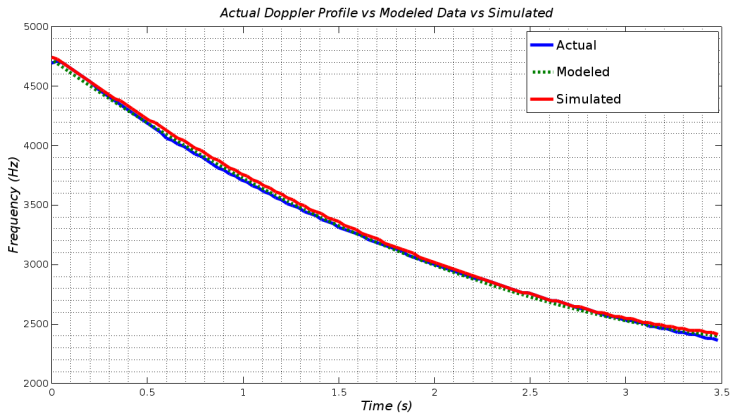


Figure 6.12: The plot of the simulated Driver shot versus the modeled and real data

Wedge. The data was captured and observed in Octave. It can be observed in Figure 6.14 that there seems to be a second trajectory above the main Pitching Wedge trajectory. As with the 6 Iron golf shot, this was accounted for as an intermodulation product that is inherent with multiplying signals. This harmonic is clearly a lot weaker than the main trajectory as it only appears for 1 second on the spectrogram.

It can be observed in the figures, that there is a lower speed and a higher deceleration compared to the 6 Iron and Driver velocity profiles. Another polynomial fit of this spectrogram plot was done to compare it to the modeled and real profile

Table 6.2: A comparison between the modeled and simulated plots relative to the real plot for the Driver golf shot

Plot	μ_{error} **(Hz)**	σ_{error} **(Hz)**
Modeled	1.95	18.28
Simulated	-26.03	19.47

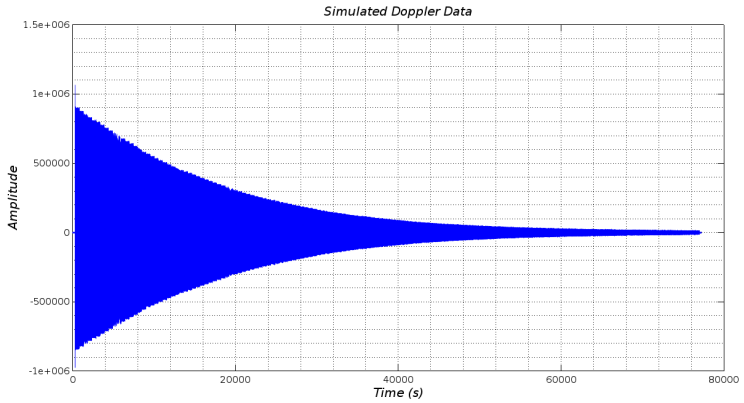


Figure 6.13: Pitching Wedge Raw Doppler

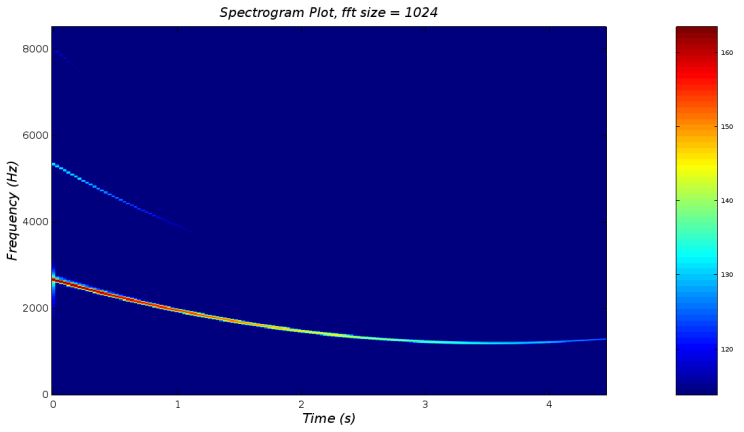


Figure 6.14: Pitching Wedge Spectrogram

captured by the Doppler radar. It can be observed in Table 6.3 that the μ_{error} has a large difference of about 22.84 Hz compared to the modeled μ_{error}. While the σ_{error} is comparable. This was seen in the previous results for the other golf shots and the reason behind this shall be discussed in the next section.

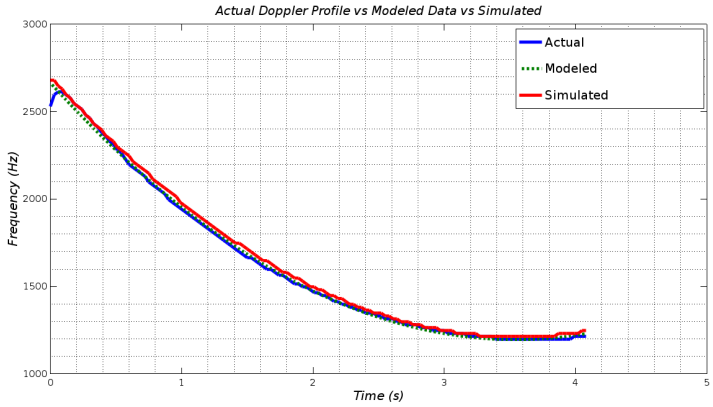


Figure 6.15: The plot of the simulated pitching wedge versus the modeled and real data

Table 6.3: A comparison between the modeled and simulated plots relative to the real plot for the Pitching Wedge golf shot

Plot	μ_{error} **(Hz)**	σ_{error} **(Hz)**
Modeled	0.47	17.82
Simulated	22.84	18.61

6.2.1 Summary of the Results

Table 6.4 shows a summary of all the shots that were simulated and compared to the modeled trajectories. It can be observed that there is a significant difference in terms of the μ_{error} between the simulated and modeled results. Although the σ_{error} seem to have a close correlation. It can be observed in the general trend of the figures that the simulated trajectories follow that of the modeled trajectories. The reason behind the offset is believed to have come from the frequency resolution. Since a sample rate of 17045 Hz was used, combined with an FFT size of 1024, a frequency resolution error of approximately 16.6 Hz can be expected.

The final observation was with regards to the amplitude decay and the second trajectory that appeared on the 6 Iron and Pitching Wedge golf shots. This was accounted for as a harmonic that is inherent with multiplying signals. It useful to note that the Driver did not have this harmonic. This can be linked to the fact

that the simulated Doppler data of the Driver had the lowest signal amplitude. This means the harmonic was too weak and was most likely suppressed by the normalization of the spectrogram.

Table 6.4: A table comparing the average error and standard deviation for all three shots

	μ_{error} **(Hz)**		σ_{error} **(Hz)**	
	Modeled	**Simulated**	**Modeled**	**Simulated**
Driver	1.951	-26.035	18.290	19.474
6 Iron	-0.489	-33.014	19.780	21.930
Pitching Wedge	0.476	22.842	17.828	18.611
Average	0.646	-12.069	18.633	20.005

6.3 Chapter Summary

This chapter covered the evaluation of the Due Doppler simulator. It first focused on evaluating the device under a controlled environment using RF test equipment. This showed that it matched the same performance of the AD9 Doppler simulator. It then showed how the device was evaluated with a Doppler radar by simulating three different types of golf shots. The final comparison of results showed the average error of the simulation compared to the real trajectories.

Chapter 7

Results And Findings

This dissertation began by highlighting a need for a Doppler simulator to perform effective HIL simulations for sport Doppler radars. It then categorized design requirements for this device in order to meet the HIL simulation standards. This device, known as the Due Doppler simulator, was then designed, developed, and evaluated based on the design requirements. These requirements stipulated that the desired Doppler simulator should operate like the existing AD9 Doppler simulator and should have certain improved features. These features included simulating various velocity profiles of golf ball trajectories, performing waveform modulation, and catering for wireless communication. These features were designed in Chapter 4, implemented in Chapter 5, and evaluated in Chapter 6. This chapter will provide a summary of the key results and findings that took place with the dissertation.

7.1 Due Doppler Simulator Integration

The first three design requirements stated that the upgraded Doppler simulator needed to function in the same way as the AD9 Doppler simulator. Furthermore, the existing housing, RF board, and antenna needed to be used. The Arduino Due microcontroller was used to upgrade the AD9 Doppler simulator to become

the Due Doppler simulator. It was integrated into the housing and operated the existing components. Furthermore, the firmware was designed with the DDS method to generate analog signals. It was observed that using the Arduino Due meant that approximately 50 % of the physical space was saved in the housing since it used less components. The electronic build time was also reduced as the Arduino Due is a complete stand-alone component. The only build time that was required was for the accessory board that housed the LPFs, HPFs, and the Bluetooth module.

7.2 Simulated Velocity Profile Analysis

The next two design requirements were specific to the main objective of this dissertation which was to simulate the velocity profile of a golf ball trajectory and perform waveform modulation. This was evaluated in the previous chapter which can be summarized by the table below. The frequencies reported in the previous chapter was converted to radial velocity.

Table 7.1: A table comparing the velocity average error and standard deviation for all three shots

	μ_{error} **(m/s)**		σ_{error} **(m/s)**	
	Modeled	**Simulated**	**Modeled**	**Simulated**
Driver	0.027	-0.364	0.256	0.273
6 Iron	-0.007	-0.462	0.277	0.307
Pitching Wedge	0.007	0.320	0.250	0.261
Average	0.009	-0.169	0.260	0.280

It can be observed that the error between real versus the simulated velocity profiles were within the design requirement of ±0.5 m/s. The average error over the three was calculated as -0.16 m/s and the standard deviation was calculated as 0.28 m/s. This is despite the fact the simulated velocity profile did not match the modeled velocity profile exactly. As mentioned earlier, the proposed reason for the velocity error is due to the frequency resolution that was used which was approximately 16.6 Hz or 0.23 m/s. This would account for the error between the modeled and simulated velocity profile. A larger FFT size could be used to

reduce the error. In terms of the frequency, amplitude, and phase modulation, this was directly used to simulate the golf ball trajectories. Even though the amplitude modulation did not play a direct role in the analysis, it still served as an reasonable representation of the signal decay of the golf ball in flight. Furthermore, the phase modulation was used to generate a phase offset between the I and Q signals. It was shown that by tuning the phase of the IQ signals slightly less then 90°, the USB could be suppressed by 40 dB relative to the LSB. It was also shown that the frequency, amplitude, and phase could be controlled via serial communication.

The final three design requirements referred to the external power supply to make the device portable, as well as communicating with it wirelessly. The budget of the upgrade was also a consideration. These requirements were achieved and documented in the design and integration phase of the dissertation. A USB power bank was used to power the device and a Bluetooth module was added to the Arduino Due so it could be operated via a smartphone. Lastly, the feasibility study showed the significant price difference between the AD9854 and the SAM3X8E.

7.3 Device Feasibility

The main finding for this dissertation was how feasible the Arduino Due was for the upgrade. The feasibility study argued the case for the Arduino Due as a suitable replacement for the AD9854 processor. This was based on its specifications that would be able to meet the design requirements, and that it has been used in generating firmware based DDS signals before. Furthermore, the significant price difference and ease of integration made it an attractive candidate. It was also indicated that the AD9854 was designed for high-speed applications whereas the projectiles that needed to be simulated were relatively low in speed. One final feature that was useful, was that the Due Doppler simulator was able to tune the IQ signals by adjusting the phase to suppress the USB. This was done digitally and without any phase-shifter board.

7.4 Chapter Summary

This chapter aimed to highlight the key results and findings that came from this dissertation. It mentioned how it was upgraded into the existing AD9 Doppler simulator and that it saved space, component usage, and component costs. It then discussed the main successful feature of the Due Doppler simulator, that was to simulate velocity profiles of a golf ball projectiles. The final section discussed the feasibility of the device, and mentioned how the Arduino Due was a suitable candidate to upgrade the AD9 Doppler simulator.

Chapter 8

Conclusions And Recommendations

This chapter is split into two sections. The first section will discuss improvements and optimization in electronic hardware, RF design, and signal processing. The second section will discuss the conclusion which will highlight the device performance, contributions, and further research.

8.1 Future Considerations

The Due Doppler simulator can be regarded as prototype that was able to simulate velocity profiles of golf ball projectiles. The design and development of this device comprised of a variety of engineering disciplines. The aim of this section is to highlight these disciplines and suggest certain considerations for further improvements and optimizations.

8.1.1 Electronic Hardware Improvements

The main electronic hardware upgrades that were performed, involved integrating the Arduino Due and peripheral components into the AD9 Doppler simulator. Furthermore, the Due Doppler simulator met all the design requirements that was stated in Chapter 1. However, in order to optimize the performance of the device, certain key features should be considered.

The first consideration regards the passive filters. A first order LPF was designed and implemented to smooth out the DAC output. It can be investigated to implement a second order low pass filter to provide a better frequency response. More importantly, the resistor used in the LPF stipulates the source current that is supplied to the RF mixer. We showed that the IF mixer is limited to a 3mA source input. By using a 2.2 V DAC, the source input became 14.6mA which was above the mixer's source input limit. Subsequently the DAC voltage had to be reduced to 0.4 V so that it would be within the mixer specifications. The simple solution to maintaining the DAC voltage resolution is to use a larger resistor with a smaller capacitor. This was not critical to the simulation of the velocity profiles but it may become important if the cutoff frequency needs to be changed for higher velocities.

8.1.2 Firmware Improvements

The firmware improvements are closely related to how the golf ball projectiles are simulated. The velocity and amplitude LUT tables can be modified to adjust for other types of trajectories. The method that was implemented used LUT tables. This is not the only way, since real-time flight models can be generated. This would allow more flexibility with the type of projectile that are simulated compared to the fixed projectiles that the Due Doppler simulator generated.

The sampling rate of the DDS can also be increased for applications that have a higher velocity. The highest sampling rate that was achieved was 500 kHz, although the DAC is rated to achieve 2 MHz. It is possible that other timers

may be able to achieve the higher sampling rate on the DAC.

8.1.3 Signal Processing Optimization

The core part of the Due Doppler simulator revolved around signal processing. As discussed in Chapter 4, raw Doppler data was measured and velocity flight models were derived for three difference types of golf shots by tracking the largest peaks in a spectrogram. This method can be optimized. Firstly, it was observed that the entire velocity profile could not be tracked as the signal become embedded in noise. Furthermore, when normalization was done, it removed some of the trajectory as well. This approach can be optimized in terms of how the spectrogram was implemented. The FFT size, overlap, and type of window can be tuned to achieve better tracking. Also, the normalization technique can be adjusted to only remove the noise so the full track can be detected.

Another process that can be optimized is how the velocity and amplitude profiles were modeled. As discussed in Chapter 4, a second order polynomial fit was chosen to represent the velocity profile. Other polynomials can be implemented to achieve a more accurate fit. Besides this empirical approach, a theoretical model could also be implemented as shown in the literature review. This could include various other parameters like drag, lift, spin, and the RCS of the ball. This would allow for more flexibility with the types of shots and projectiles that could be simulated.

The amplitude decay profile can also be optimized. Only the data from after the gain switch was used. A two-stage amplitude decay profile could be simulated to simulate what happens in reality. Lastly, only the profile of the ball was simulated. It was observed in Chapter 4 that a club profile could be simulated as well.

8.1.4 RF Design Updates

The one investigation that should take place is to determine the cause of the 50 Hz offset. This would allow for a more accurate simulated velocity profile that would match the modeled profile. Furthermore, the FlightScope Doppler radars measure angles as well. This can be a scope for further research to simulate the change in azimuth and elevation angles of the golf ball trajectory. This would possibly require modifications to the RF design.

8.2 Conclusion

A Doppler simulator is a fundamental HIL device that is used to evaluate Doppler radars. This dissertation focused on developing a Doppler simulator that would be able to do real-time velocity and amplitude simulations of golf ball trajectories. This was achieved by upgrading a Doppler simulator with an Arduino Due that could simulate velocity profiles of golf ball trajectories. The DDS method was used to generate IQ signals that was modified to implement flight models derived from empirical golf ball data. Furthermore, a simulated velocity average error of -0.16 m/s and a standard deviation of 0.28 m/s was achieved for the three types of golf shots.

8.2.1 Contributions

This dissertation was able to develop a device to perform a HIL simulation. This was done by upgrading an existing Doppler simulator that could only simulate a single velocity. It was shown that the Doppler simulator achieved its objective and met the design requirements. Furthermore, this device is unique in the sense that it used a consumer based microcontroller to perform effective HIL simulations. The application of this device is also novel, as very little research has been done for sport Doppler radars let alone sport Doppler simulators.

8.2.2 Future Research

The scope for future research for the Due Doppler simulator is vast. It can be implemented in a variety of applications specifically focusing on simulating the velocity and amplitude profile of a projectile. In the field of golf, the simulation of the full 3D trajectory is an intriguing concept to simulate as this would allow the FlightScope Doppler radar to be fully evaluated. Furthermore, the capability of the firmware based DDS implementation allows for multiple targets to be generated that could be moving towards or away from the radar. The applications are almost endless, the only limitation is the imagination.

Bibliography

[1] T. I. V. B. Johansen, T. A.; Fossen, "Hardware-in-the-loop testing of dp systems." *DP Conference*, 2005.

[2] J. J. Strydom, J. E. Cilliers, M. Gouws, D. Naicker, and K. Olivier, "Hardware in the loop radar environment simulation on wideband drfm platforms," in *IET International Conference on Radar Systems (Radar 2012)*, Oct 2012, pp. 1–5.

[3] R. Lefevre, R. Durand, R. Satterfield, A. MacMullen, J. Walker, and G. Pollon, "Smart repeater for radar testing," in *1998 IEEE AUTOTESTCON Proceedings. IEEE Systems Readiness Technology Conference. Test Technology for the 21st Century (Cat. No.98CH36179)*, Aug 1998, pp. 611–614.

[4] J. J. Strydom, J. J. de Witt, and J. E. Cilliers, "High range resolution x-band urban radar clutter model for a drfm-based hardware in the loop radar environment simulator," in *2014 International Radar Conference*, Oct 2014, pp. 1–6.

[5] D. Kim, H. S. Oh, and S. W. Hwang, "A dds-based distributed simulation for anti-air missile systems," in *2016 6th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH)*, July 2016, pp. 1–7.

[6] F. Chen, R. Li, L. Ding, L. Liu, L. Dai, and G. Deng, "A method against drfm dense false target jamming based on jamming recognization," in *IET International Radar Conference 2015*, Oct 2015, pp. 1–4.

[7] *K-DT1 Radar Doppler Target*, RFbeam Microwave GmbH, Farbgutstrasse 3 CH-9008 St.Gallen Switzerland, handheld Doppler Simulator.

[8] J. Hu, F. Wang, N. Cao, and Z. Li, "A smart repeater for weapon location radars based on time-frequency analysis," in *2009 35th Annual Conference of IEEE Industrial Electronics*, Nov 2009, pp. 3349–3352.

[9] L. Xing-hai, Y. Jian, Y. Yong, and S. Kai, "Design of wideband radar signal simulator based on high-speed dds technology," in *2016 IEEE International Conference on Signal and Image Processing (ICSIP)*, Aug 2016, pp. 548–551.

[10] F. Trahan, "Golf playing field with ball detecting radar units," Jun. 16 1987, uS Patent 4,673,183. [Online]. Available: https://www.google.com/patents/US4673183

[11] A. Dilz, "Miniature sports radar speed measuring device," Jan. 26 1999, uS Patent 5,864,061. [Online]. Available: https://www.google.com/patents/US5864061

[12] R. McNeal and M. Belkhayat, "Standard tools for hardware-in-the-loop (hil) modeling and simulation," in *2007 IEEE Electric Ship Technologies Symposium*, May 2007, pp. 130–137.

[13] (2017) Flightscope technology. [Online]. Available: https://flightscope.com/company/technology/

[14] Z. Peng, "Realization of drfm radar target simulator based on general instruments," in *IET International Radar Conference 2015*, Oct 2015, pp. 1–8.

[15] Arduino. (2013, June) Arduin due, store. [Online]. Available: https://store.arduino.cc/arduino-due/

[16] Scilab. (2017, Feb) Scilab 6.0.0. [Online]. Available: http://www.scilab.org/en/download/latest

[17] *Octave v4.2.1*, GNU Octave, 2017.

[18] Arduino. (2017, Aug.) Arduino v1.8.4. [Online]. Available: https://www.arduino.cc/en/Main/Software

[19] Fritzing. (2016, June) Fritzing v0.9.3b. [Online]. Available: http://fritzing.org/home/

[20] M. Richards, W. Holm, and J. Scheer, *Principles of Modern Radar: Basic Principles*, ser. Electromagnetics and Radar. Institution of Engineering and Technology, 2010. [Online]. Available: https://books.google.co.za/books?id=nD7tGAAACAAJ

[21] R. Thomas. (2003, Dec.) Doppler detectives. [Online]. Available: https://plus.maths.org/content/doppler-detectives

[22] B. Muro. (2008, Dec.) Characterizing radar interference immunity. [Online]. Available: http://www.noisecom.com/resource-library/articles/characterizing-radar

[23] C. Nickolas. (2011, Oct.) The basics of mixers. [Online]. Available: https://www.digikey.com/en/articles/techzone/2011/oct/the-basics-of-mixers

[24] *IQ, Image Reject, and Single-Sideband Mixers*, Marki Microwave, jun 2013.

[25] B. Math. (2017) Sum and difference formulas. [Online]. Available: https://brownmath.com/twt/sumdiff.htm

[26] *What is the deal with IP2 in mixers?*, Marki Microwave, oct 2014.

[27] R. Lyons. (2013, April) A quadrature signals tutorial: Complex, but not complicated. [Online]. Available: https://www.dsprelated.com/showarticle/192.php

[28] M. Christiano. (2015, Nov.) Everything you need to know about direct digital synthesis. [Online]. Available: https://www.allaboutcircuits.com/technical-articles/direct-digital-synthesis/

[29] L. SINE. (2008, Oct.) Fundamentals of direct digital synthesis (dds). [Online]. Available: www.analog.com/media/en/training-seminars/tutorials/MT-085.pdf

[30] L. Cordesses, "Direct digital synthesis: a tool for periodic wave generation (part 1)," *IEEE Signal Processing Magazine*, vol. 21, no. 4, pp. 50–54, July 2004.

[31] B. Fugerer. (2017, feb) Embedded design ideas applying direct digital synthesis with a twist. [Online]. Available: https://occamtechgroup.com/2016/02/26/embedded-design-ideas-applying-direct-digital-synthesis-with-a-twist/

[32] D. B. (2012, Dec.) Arduino due dds - part 1 - sinewaves and fixed point maths. [Online]. Available: http://rcarduino.blogspot.co.za/2012/12/arduino-due-dds-part-1-sinewaves-and.html

[33] L. Cordesses, "Direct digital synthesis: a tool for periodic wave generation (part 2)," *IEEE Signal Processing Magazine*, vol. 21, no. 5, pp. 110–112, Sept. 2004.

[34] E. Colombini. (2014, Oct.) How a pwm dac works. [Online]. Available: http://www.quintadicopertina.com/enricocolombini/2015/05/03/how-a-pwm-dac-works/

[35] D. Marshall. (2001, April) Nyquist's sampling theorem. [Online]. Available: https://users.cs.cf.ac.uk/Dave.Marshall/Multimedia/node149.html

[36] ThomasGolf.com. (2016, April) Golf terms. [Online]. Available: http://golf-info-guide.com/golf-terms/

[37] R. Barber. (2007, May) Golf ball flight dynamics. [Online]. Available: rancebarber.com/resources/AEP434FinalProject_GolfBallDynamics.pdf

[38] J. Dr. John C. Adams. (2012, May) Golf ball flight trajectory in time: 4th-order runge-kutta numerical integration. [Online]. Available: www.physics.usyd.edu.au/teach_res/excel/golfball.xls

[39] Analog.com. Ad9854 cmos 300 msps quadrature complete dds. [Online]. Available: www.analog.com/media/en/technical-documentation/data-sheets/AD9854.pdf

[40] *What is an Arduino*, Sparkfun, 2017.

[41] M. Margolis, *Arduino Cookbook*. O'Reilly Media, Inc., 2011.

[42] M. Nawrath. (2009, Nov.) Arduino dds sinewave generator. [Online]. Available: http://interface.khm.de/index.php/lab/interfaces-advanced/arduino-dds-sinewave-generator/

[43] S. Purohit. (2016, Nov.) How to make an audio player with speaker using the arduino uno! [Online]. Available: https://diyhacking.com/arduino-audio-player/

[44] U. Sear. (2016, Nov.) Arduino based dds signal generator using ad9851 lgpl. [Online]. Available: https://create.arduino.cc/projecthub/umar-sear/arduino-based-dds-signal-generator-using-ad9851-ed4d8e

[45] Adafruit. (2017, March) Mcp4725 breakout board - 12-bit dac w/i2c interface. [Online]. Available: https://www.adafruit.com/product/935

[46] M. B. Arturo Guadalupi, Scott Fitzgerald. (2015, Dec.) How to make an audio player with speaker using the arduino uno! [Online]. Available: https://www.arduino.cc/en/Tutorial/SimpleAudioPlayer

[47] L. Modes. (2015, March) Sam3x / sam3a series - atmel — smart arm-based mcu. [Online]. Available: http://www.atmel.com/Images/Atmel-11057-32-bit-Cortex-M3-Microcontroller-SAM3X-SAM3A_Datasheet.pdf

[48] *AD9854ASVZ, Direct Digital Synthesizer 12 bit-Bit 300Msps, 3.135 3.465 V, 80-Pin TQFP*, RS Components, 2017.

[49] *Microchip ATSAM3X8EA-AU, 32bit ARM Cortex M3 Microcontroller, 84MHz, 512 kB Flash, 144-Pin LQFP*, RS Components, 2017.

[50] *Arduino Due*, RS Components, 2017.

[51] Analog. (2017) Hmc521. [Online]. Available: www.analog.com/media/en/technical-documentation/data-sheets/hmc521.pdf

[52] I. Modelithics. (2017) Nbb-300. [Online]. Available: https://www.modelithics.com/models/Vendor/Qorvo/NBB-300.pdf

[53] J. O. S. III. (2017) Spectral audio signal processing - chapter 10 - spectrogram. [Online]. Available: https://www.dsprelated.com/freebooks/sasp/

[54] *specgram (x)*, GNU Octave, 2017.

[55] *28.5 Polynomial Interpolation*, GNU Octave, 2017.

[56] bmj. (2017) 2. mean and standard deviation. [Online]. Available: http://www.bmj.com/about-bmj/resources-readers/publications/statistics-square-one/2-mean-and-standard-deviation

[57] D. Boschen. (2012, Nov.) Numercially controlled oscillator. [Online]. Available: https://dsp.stackexchange.com/questions/31540/problem-in-understanding-ddfs-direct-digital-frequency-synthesizer

[58] Olimex. (2017) Bluetooth-serial-hc-06. [Online]. Available: https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf

[59] K. Morich. (2017, July) Serial bluetooth terminal v1.9. [Online]. Available: https://play.google.com/store/apps/details?id=de.kai_morich.serial_bluetooth_terminal

Appendix A

Datasheets

Pin Descriptions

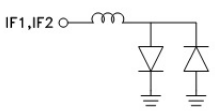
Pin Number	Function	Description	Interface Schematic
1, 2, 6 - 8, 10, 13, 17 - 24	N/C	No connection required. These pins may be connected to RF/DC ground without affecting performance.	
3, 5, 12, 14, 16	GND	These pins and package bottom must be connected to RF/DC ground.	
4	RF	This pin is AC coupled and matched to 50 Ohms from 8.5 to 13.5 GHz.	
9	IF1	This pin is DC coupled. For applications not requiring operation to DC, this port should be DC blocked externally using a series capacitor whose value has been chosen to pass the necessary IF frequency range. For operation to DC, this pin must not source/sink more than 3mA of current or part non-function and possible part failure will result.	
11	IF2		
15	LO	This pin is AC coupled and matched to 50 Ohms from 8.5 to 13.5 GHz.	

Figure A.1: RF Mixer Pin Description

44.6 Functional Description

44.6.1 Digital-to-Analog Conversion

The DACC uses the master clock (MCK) divided by two to perform conversions. This clock is named DACC Clock. Once a conversion starts the DACC takes 25 clock periods to provide the analog result on the selected analog output.

Figure A.2: SAM3X8E DAC Characteristics

45.2 DC Characteristics

The following characteristics are applicable to the operating temperature range T_A = -40°C to 85°C, unless otherwise specified.

Table 45-2. DC Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{DDCORE}	DC Supply Core		1.62	1.8	1.95	V
V_{DDIO}	DC Supply I/Os		1.62	3.3	3.6	V
V_{DDBU}	Backup I/O Lines Power Supply		1.62		3.6	V
$V_{DDUTMII}$	USB UTMI+ Interface Power Supply		3.0		3.6	V
V_{DDPLL}	PLL A, UPLL and Main Oscillator Supply		1.62		1.95	V
V_{DDANA}	ADC Analog Power Supply		(1)		(1)	V
V_{IL}	Input Low-level Voltage	PIOA/B/C/D/E/F[0–31]	-0.3		0.3 × V_{DDIO}	V
V_{IH}	Input High-level Voltage	PIOA/B/C/D/E/F[0–31]	0.7 × V_{DDIO}		V_{DDIO} + 0.3V	V
V_{OH}	Output High-level Voltage	PIOA/B/C/D/E/F[0–31] I_{OH} ~ 0 I_{OH} > 0 (See I_{OH} details below)	 V_{DDIO} - 0.2V V_{DDIO} - 0.4V			V
V_{OL}	Output Low-level Voltage	PIOA/B/C/D/E/F[0–31] I_{OH} ~ 0 I_{OH} > 0 (See I_{OL} details below)			 0.2 0.4	V
V_{hys}	Hysteresis Voltage	PIOA/B/C/D/E/F[0–31] except PA0, PA9, PA26, PA29, PA30, PA31, PB14, PB22, PC[2–9], PC[15–24], PD[10–30], PE[0–4], PE15, PE17, PE19, PE21, PE23, PE25, PE29	150		500	mV
		ERASE, TST, FWUP, JTAGSEL	230		700	mV
I_{OH}	Source Current	1.62V < VDDIO < 1.95V; V_{OH} = V_{DDIO} - 0.4 - Group 1(2) - Group 2(3)			 -8 -3	mA
		3.0V < VDDIO < 3.6V; V_{OH} = V_{DDIO} - 0.4 - Group 1(2) - Group 2(3)			 -15 -3	
		1.62V < VDDIO < 3.6V; V_{OH} = V_{DDIO} - 0.4 - NRST, TDO			 -2	
		Relaxed Mode: 3.0V < VDDIO < 3.6V; V_{OH} = 2.2V - Group 1(2) - Group 2(3)			 -24 -9	

Figure A.3: SAM3X8E DC DAC characteristics part 1

Table 45-41. Channel Conversion Time and DAC Clock

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{DAC}	Clock Frequency		1		50	MHz
t_{CP_DAC}	Clock Period		20		1000	ns
f_S	Sampling Frequency		0.04		2	MHz
t_{START}	Startup time	From Sleep Mode to Normal Mode: - Voltage Reference OFF(1) - DAC Core OFF	23	34	45	µs
		From Fast Wake-up to Normal Mode: - Voltage Reference ON - DAC Core OFF	2.5	4	5	
t_{CONV}	Conversion Time			25		t_{CP_DAC}

Notes: 1. External voltage reference for DAC is ADVREF. See the ADC voltage reference characteristics in Table 45-31, "External Voltage Reference Input," on page 1403.

Figure A.4: SAM3X8E DC DAC characteristics part 2

Appendix B

Doppler Simulator Specifications

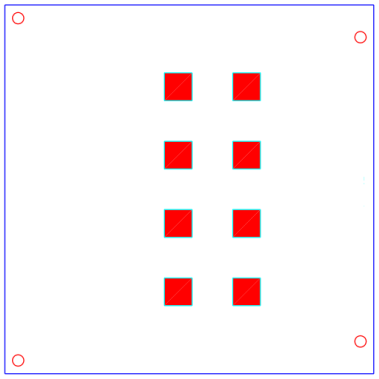


Figure B.1: An illustration for the pattern of the patch array antenna.

Specification	**Value**
Antenna	
Frequency Band	10.1 - 10.9 GHz
Doppler Frequency Band	1 - 20 000 Hz
Array Pattern	4x2 Patches
Polarization	Horizontal
Azimuth Beamwidth	20°
Elevation Beamwidth	40°
Antenna Gain	14 dBi
RF Mixer	
IF Bandwidth	DC - 3.5Ghz
Image Rejection	38 dB
LO to RF isolation	50 dB
Operation	IQ Single Sideband Upconverter
DDS	
Clock Frequency	300 MHz
DAC Resolution	12 bit
Phase accumulator resolution	48 bit
Sample rate	20 MSps
Accessory Board	
Supply Voltage	12 V
Supply Current	350 mA
Communication Link	RS 232
Baud Rate	115 200 bps

Table B.1: A table of specifications that is used in the Doppler Simulator

Appendix C

Doppler Simulator Testing

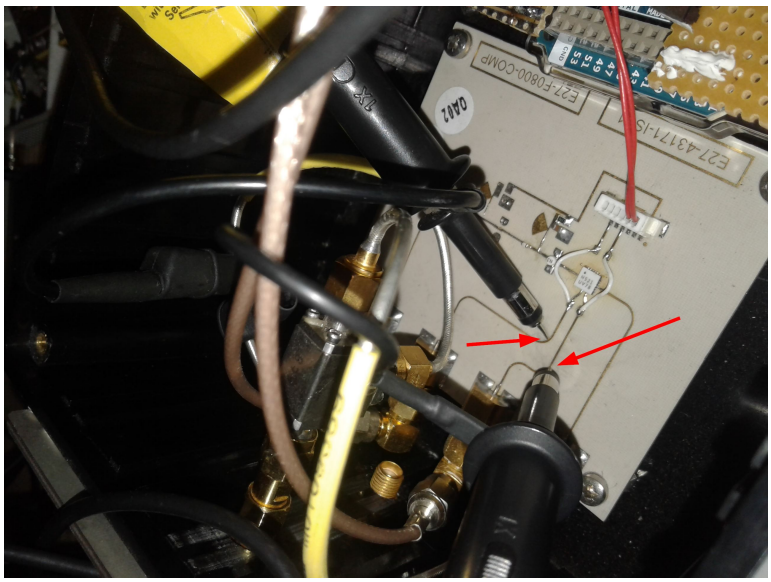


Figure C.1: Test points on the RF board showing where the IQ signals were measured.

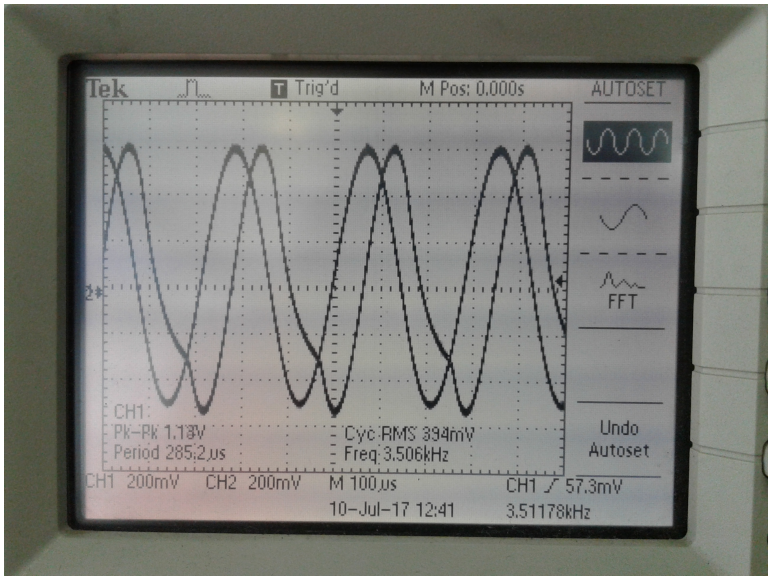


Figure C.2: Using an oscilloscope this shows a frequency spectrum of the two IQ signals showing 3.5 kHz. The discontinuity in the lagging signal is caused by the phase shifter board to create a slight phase adjustment to suppress the USB.

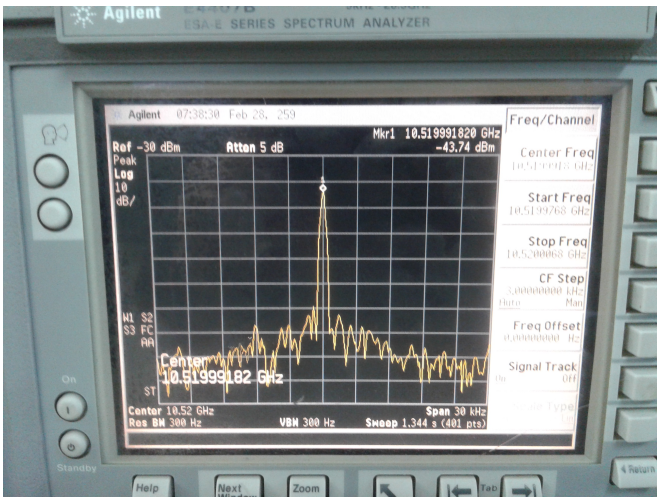


Figure C.3: A snapshot of the spectrum analyzer for the AD9 Doppler simulator while the X-band antenna is transmitting.

Appendix D

Bill of Materials

Bill of Materials: ArduinoDue_v3.fzz

Assembly List

Label	Part Type	Properties
C3	Ceramic Capacitor	voltage 6.3V; capacitance 1µF; package 100 mil [THT, multilayer]
C4	Ceramic Capacitor	voltage 6.3V; capacitance 1µF; package 100 mil [THT, multilayer]
C5	Ceramic Capacitor	voltage 6.3V; capacitance 100nF; package 100 mil [THT, multilayer]
C6	Ceramic Capacitor	voltage 6.3V; capacitance 100nF; package 100 mil [THT, multilayer]
R3	330Ω Resistor	pin spacing 400 mil; package THT; tolerance ±5%; bands 4; resistance 330Ω
R4	330Ω Resistor	pin spacing 400 mil; package THT; tolerance ±5%; bands 4; resistance 330Ω
R5	100Ω Resistor	pin spacing 400 mil; package THT; tolerance ±5%; bands 4; resistance 100Ω
R6	100Ω Resistor	pin spacing 400 mil; package THT; tolerance ±5%; bands 4; resistance 100Ω
U1	Arduino Due (Rev2b)	type Arduino Due
U2	Bluetooth HC-06 Macho	variant variant 4; protocol Bluetooth

Shopping List

Amount	Part Type	Properties
2	Ceramic Capacitor	voltage 6.3V; capacitance 1µF; package 100 mil [THT, multilayer]
2	Ceramic Capacitor	voltage 6.3V; capacitance 100nF; package 100 mil [THT, multilayer]
2	330Ω Resistor	pin spacing 400 mil; package THT; tolerance ±5%; bands 4; resistance 330Ω
2	100Ω Resistor	pin spacing 400 mil; package THT; tolerance ±5%; bands 4; resistance 100Ω
1	Arduino Due (Rev2b)	type Arduino Due
1	Bluetooth HC-05 Macho	variant variant 4; protocol Bluetooth

Figure D.1: The bill of materials for the Due Doppler simulator.

Appendix E

RF Test Equipment

The Doppler simulator was tested in the RF lab at FlightScope. Various RF test equipment was used to test and evaluate the functionality of the Doppler simulator at various points. The first test equipment that was used was a Tektronix TBS1202B 200 MHz 2 channel oscilloscope. This was mainly used to test the I and Q signals generated by the two DACs.

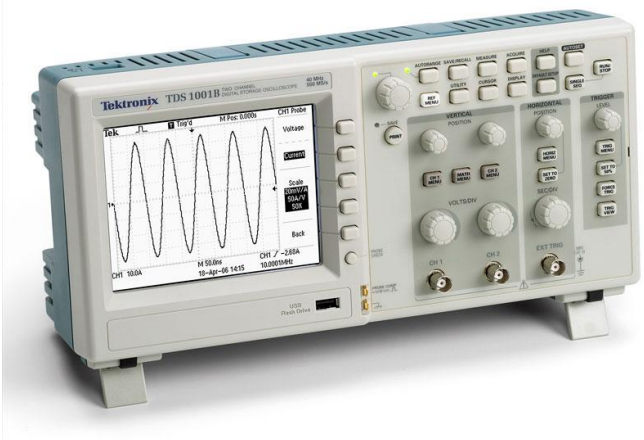


Figure E.1: The Tektronix TDS1002B Oscilloscope.

The other test equipment that was used was an Agilent E4407B spectrum analyzer with a range of 9 kHz to 26.5 GHz. This was quite a crucial piece of equipment to test the general functionality of the RF components, and whether

the correct sidebands were being generated from the mixer. And image of this is shown in Figure E.2.

Figure E.2: The Agilent E4407B Spectrum Analyzer.

To test the Doppler simulator, an X-band horn antenna was attached to the spectrum analyzer and another connected to a 10.5 Ghz frequency synthesizer. A 50 Ω SMA coaxial cable was used to connect the receiver horn antenna to the spectrum analyzer as well as connecting the frequency synthesizer to the transmitter horn antenna. In line with the spectrum analyzer was an Agilent blocking capacitor and in line with the transmitter horn antenna was a 20 dB attenuator.

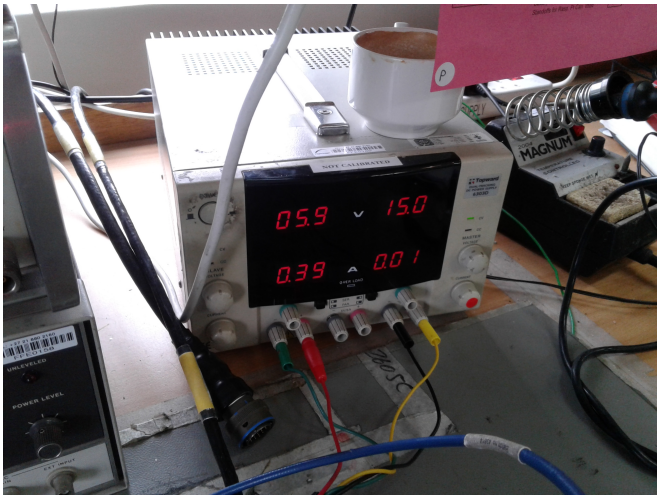


Figure E.3: The Power Supply

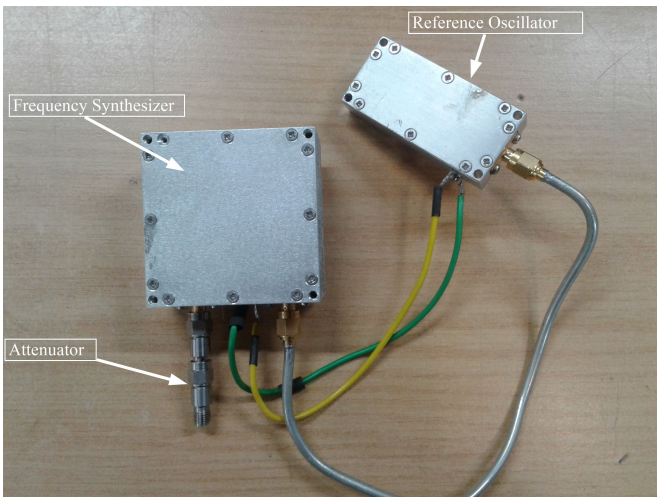


Figure E.4: The frequency synthesizer and reference oscillator

By first pointing these two horns together, a 10.5 Ghz peak on the spectrum analyzer could be seen. Then the horn antennas were both pointed to the Doppler simulator at 3 m to simulate the actual application of the FlightScope Doppler radar.

Figure E.5: The two X-band horn antennas

Some testing was done in a makeshift anechoic chamber used at the FlightScope RF lab, however the signal interference from the lab surroundings provided little overall effect to testing the Doppler simulator.