Study of the TR Synchronization and Video Conversion Unit

Justin Kuruvilla Abraham

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

Cape Town, August 2012

Declaration

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

Signature of Author .....

Cape Town
10 August 2012

Abstract

This dissertation describes the design and testing of a model of the Synchronization and Video Conversion Unit (SVCU), a subsystem of the tracking radar (TR) at Denel Overberg Test Range (OTR). The SVCU synchronizes all the radar sub-systems and also converts the returned RF target signals to digital numbers. The technology within the SVCU is outdated and spares are scarce if not unattainable. This study forms the first phase of the development of a new SVCU and will determine the specifications of the hardware needed to build the replacement.

Models of the transmit and receive chain of the radar were first developed in SystemVue^{TM}. A comprehensive literature review was then done, yielding an accurate model of the current SVCU. The radar model was run, with simulated target and scene parameters, and its output fed into the SVCU model. The output of the SVCU was then processed by a CFAR detector and gated tracking algorithms implemented in MathLang and Python. The simulated target was correctly identified in the range-Doppler plane.

The tracking gates (used to measure range and Doppler) were then corrupted with jitter, rise-time and offsets. A statistical analysis was done on the effect of these impurities on the radar measurements. A new SVCU architecture, utilizing high speed ADCs and digital integrators, was then tested. The effects of non-linearities (DNL and INL) in the ADC and phase noise on the ADC sample clock on the radar measurements were analysed.

The jitter on the transmit sync (TX), the ADC sample clock and tracking gates were found to be the most critical aspects of the SVCU. To meet the specified measurement accuracy of the radar, the root-sum-square of the jitter on these syncs (jitter budget) must not exceed 30 nanoseconds.

A case study was then done to determine the jitter budget achievable in an FPGA-centric SVCU design. The study concluded that a jitter budget of 30 ns is achievable. Moreover, in an FPGA based design the jitter introduced by the interface sending the TX sync from the FPGA (SVCU) to the transmitter assembly will, almost entirely, determine the range accuracy of the TR.

From these findings, a new SVCU, based on the RHINO board from the UCT RRSG, was recommended and the future work outlined.

Contents

List of Figures

List of Tables

List of Symbols

B	—	RF bandwidth of the transmitted signal
c	—	Speed of light
ε_D	—	The error between the estimated and actual target Doppler frequency
ε_R	—	The error between the estimated and actual target range
f_c	—	Radar centre carrier (transmit) frequency
f_{prf}	—	Pulse repetition frequency
λ	—	Wavelength
K	—	Number of filling pulses in a radar frame
L	—	Number of pulses in a radar frame
N	—	Number of active pulses in a radar frame
P_{avg}	—	Average transmitted power
P_t	—	Peak transmitted power
R	—	Slant range to target or object of interest
RF_l	—	Radar frame length in seconds
τ	—	Pulse width
T_R	—	The time taken from transmission of the RF signal to reception of its echo
v_r	—	Radial velocity of a target towards the radar

Nomenclature

1553 MUXBUS — A serial bus communication protocol based on the military specification 1553-B.

A-scope — A deflection modulated display in which the vertical deflection is proportional to target echo strength and the horizontal coordinate is proportional to range.

A/D — Analogue to digital.

Active pulse — A pulse (see PRI) that contains a target echo.

ADC — Analogue to digital converter, an electronic device that converts an input analogue voltage or current to a digital number proportional to the magnitude of the voltage or current.

Azimuth — The angle between a horizontal reference direction (usually north) and the horizontal projection of the direction of interest, usually measured clockwise.

Beacon mode — Tracking of targets carrying a standard transponder (non-coherent) in "C" band.

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.

CCA — Command and Control Assembly. The interface between the operator and the TR, operators' station.

CFA — Cross-field amplifier. A microwave tube in which the output amplification results from the interaction of the electromagnetic wave propagating along the slow-wave circuit and the electron beam moving in crossed electric and magnetic fields.

Coherence — Coherency in a radar is the ability of the radar to preserve and extract the phase shift, due to the motion of a target, from the received echo signal. This phase information can then be used to calculate the velocity of the target.

COTS — Commercially available Off-The-Shelf.

CPI — Coherent processing interval.

CW — Continuous wave.

D/A — Digital to analogue.

DAC — Digital to analogue converter, a device that converts a digital (usually binary) code to an analogue signal.

DFT — Discrete Fourier transform, a discrete transform used in Fourier analysis to transform a time domain function into its frequency domain representation.

DNL — Differential non-linearity, the difference between an actual step width and the ideal value of 1 LSB in an ADC.

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

Elevation — The angle between a horizontal reference plane and line of sight in the direction of interest, measured upward.

EM — Electromagnetic.

FFT — Fast Fourier transform, refers to a way the discrete Fourier Transform (DFT) can be calculated efficiently, by using symmetries in the calculated terms. The symmetry is highest when n, the number of points, is a power of 2, and the transform is therefore most efficient for these sizes.

FPGA — Field Programmable Gate Array.

HSD — High Speed Data interface. A 32 bit wide, unidirectional, bus used to transfer data between the SVCU and TR system computer.

INL — Integral non-linearity, the deviation, in LSB or percent of full-scale range, of the actual ADC transfer function from a straight line.

IRIG B — Inter-range instrumentation group time code, version B. IRIG is a telecommunications working group standard time code as defined in IRIG Standard document 200-04. The primary difference between the different versions of IRIG is the bit rate. IRIG B has a bit rate of 100 pulses per second.

MOSC — Master radar oscillator. The primary oscillator in the TR, from which all frequencies in the radar are derived.

OTR — Denel Overberg Test Range.

Pilot — A reference signal injected into the radar for test and-or calibration purposes.

PRF — Pulse repetition frequency. The number of pulses per unit of time, usually per second and expressed in Hertz (Hz).

PW — Pulse width. The width of the transmitted radar pulse.

PRI — Pulse repetition interval. The reciprocal of PRF.

Radar — Radio detection and ranging.

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

RB — Range bin.

RCS — Radar cross section. A measure of the reflective strength of a radar target, usually expressed in square metres.

RF — Radio frequency.

RFFE — RF Front End.

RFH — RF Head.

R-scope — An A-scope with a segment of the time base expanded near the target for greater accuracy in distance measurement.

RT — Real time.

SEL — SEL Concept 32/8750 "super mini-computer". This is the TR system computer and functions both as a system controller and signal processor.

SEL(x)W(y:z)B(m:s) — Denotes a 1553 MUXBUS message from the SEL to the SVCU where: x is the message number; y the word or y:z the word range; and m the bit or m:s the bit range.

SI — International System of Units.

Skin mode — Tracking of targets on the reflected signal determined by their radar cross sections.

SNR — Signal to noise ratio. The ratio of signal power S to noise power N within the receiving system bandwidth, usually expressed in decibels (dB).

SP — Signal processing. Radar signal processing is an operation in which desired data

about a target is extracted from the received radar signal.

SV — SystemVue^{TM}, an electronic design automation (EDA) environment for electronic system-level (ESL) design produced by Agilent Technologies.

SVCU — Synchronization and Video Conversion Unit.

SYNTH-C — Synthesizer C-band. C-band frequency synthesizer, referenced to the MOSC, found in the TR RFH. It is used to set the carrier frequency of the transmit pulse and down-covert the received echo in skin mode.

TR — Tracking radar. Lunette tracking radar as deployed at Denel Overberg Test Range.

TRLA — Transmit receive limiter attenuator. A device employed in the TR for attenuating the transmit energy fed back into the radar receiver, but concurrently allowing target echo signals to reach the receiver without appreciable loss.

TWT — Travelling wave tube, an electron tube in which a stream of electrons interacts continuously or repeatedly with a guided electromagnetic wave moving substantially in synchronism with it, and in such a way that there is a net transfer of energy from the stream to the wave.

TX — Transmit.

VCO — Voltage-controlled oscillator. An oscillator whose frequency varies with an applied voltage.

VPU — Vector processing unit.

Waveguide — A transmission line comprising a hollow conductive tube within which electromagnetic waves may be propagated or a solid dielectric or a dielectric-filled conductor for the same purpose.

Chapter 1

Introduction

This chapter gives a background to this dissertation, its purpose and what the objectives are. The chapter concludes with a discussion on the scope and structure of the remainder of the document.

1.1 Background to Project

In the late 1980s Denel Overberg Test Range (OTR) acquired three tracking radars (TR). The TRs are fully computer controlled systems with a SEL Concept 32/8750 super mini-computer [1].

At the heart of the tracking radar is the synchronization and video conversion unit (SVCU). The SVCU synchronizes (coordinates the operation of) all the radar sub-systems from computer, transmitter and receiver racks, right down to the scopes on the TR operators' control assembly (CCA) [2]. It also converts the returned target signals from RF (analogue) to floating point numbers (digital).

The SVCU is a complex piece of hardware, custom built for the Lunette TR. It consists of a microprocessor, analogue and digital electronics (see Figure 4.1 for a block diagram of the current SVCU). Unfortunately the SVCU is also one of the least understood and most difficult sub-systems of the TR to debug/fault find. Most of the technology within it is outdated and spares are scarce if not unattainable.

1.2 Project Inception and User Requirements

The reasons stated above warranted the development of a replacement for the SVCU. This process would require an understanding of the analogue, digital and signal processing (SP) aspects of the TR and general radar theory/principles.

In 2010 a detailed study was started into the replacement of the SVCU by Justin Abraham, an engineer at OTR (author). This study will serve to demystify the functioning of the SVCU and its purpose within the OTR tracking radar. The SVCU study would also form part of the author's MSc project and would be done in close collaboration with the Radar Remote Sensing Group (RRSG) at UCT.

1.2.1 Objectives

The primary objective of this dissertation is to build a working model of the SVCU. This is necessary to determine the specifications of the hardware needed to build the replacement. In order to achieve this, and to validate this model, the following objectives will have to be met

1. Develop a model to simulate the analogue signal flow down the TR receive chain;
2. Conduct an in depth study of the theory of operation of the SVCU;
3. Build a system level simulation of the SVCU based on the findings;
4. Investigate the signal processing algorithms within the TR system computer and implement an equivalent detector and tracker;
5. Integrate the receive chain, SVCU and signal processing models and test the system with simulated target returns;
6. Test the performance of this radar model when the time critical signals from the SVCU are 'corrupted' (jitter, drift or not synchronized within limit);
7. Propose a system level model for a new SVCU, using hardware available in industry, and perform a critical timing study thereof.

1.3 Scope and Limitations

A study and simulation of this nature poses the following limitations:

The information presented within this document stems from the development documents, output at the various design phases of the Lunette radar, and the final documentation acquired by OTR when the radar was commissioned. These documents span three decades (1984-2011), and some contradict on the functionality of the SVCU.

The scant nature of the final documentation meant that a large portion of this document is formulated from the development documents of the SVCU. Some of the functionalities that existed in the design phases may not exist in the final SVCU. This is also an unknown.

Only the 'critical' signal processing (SP) aspects, deemed relevant for this study by the author, have been investigated and presented. The SP pertaining to calibrations and the pilot signal have been cursorily shown.

All permutations (pulse width, PRF, channels etc.) of the SVCU have not been simulated. This is cumbersome and would not yield significantly more information. An ideal SVCU, with typical operating parameters as found during the operation of the TR at the test range, has been modelled.

Mathematical models of the 'target' (object of interest) and the scene (RCS, clutter, noise, atmospheric attenuation etc.) have been kept simple and as depicted in the radar texts [3] [4] [5].

The simulation software used, SystemVue^{TM}, is an environment for electronic system-level (ESL) design [6]. SystemVue^{TM}is primarily used to realise the physical layer (PHY) of wireless and aerospace/defense communications systems. The extensive RF and DSP libraries and the synchronous nature of SystemVue^{TM}means it lends itself as a good platform for the simulation of a radar. There are however inherent properties of the software (memory size, computation speed and drift, limitations of RF components etc.) that limit the characteristics of the radar model.

The findings of the study are based on a system level simulation of the TR and SVCU. For the simulation to tend towards the 'real world' system, the simulation would have to grow in complexity and size. This process can become arduous and out of the scope of this project.

The complexity and size of the simulation are thus kept at an order sufficient to extract the main characteristics of the SVCU that the study seeks.

1.4 Project Outline

This dissertation is divided into 7 chapters. Here a brief overview, and the key findings, of each chapter is presented.

1.4.1 Chapter 2: Lunette Tracking Radar

This chapter introduces the Lunette Tracking Radar (TR) to the reader. An overview of the main sub-systems of the TR is presented, to give an understanding of the transformations the target echo goes through before it is sampled by the SVCU.

The TR is a pulsed Doppler radar with specifications as per Table 1.1 [7]:

Table 1.1: The OTR tracking radar technical specifications.

Parameter	**Specification**
Transmit frequency	5400 - 5900 MHz (in 5 MHz steps)
Peak power	600 kW (TWT + CFA)
Pulse width	0.5, 1.0 or 2.5 μs
PRF	300 - 3000 Hz
Antenna gain	: 43 dB
Antenna beam width	: 1°
Maximum range	180 km (skin mode, 1 m^2 target RCS)
	2800 km (beacon mode)
Maximum velocity	10 km/sec (radial)
	42°/sec (angular)
Maximum acceleration	300 m/sec^2 (radial)
	20°/sec^2 (angular)

The TR transmitter chain is a three stage cascaded amplifier consisting of a solid state amplifier (SSA), a cross field amplifier (CFA) and a travelling wave tube (TWT) [8].

Target echo signals enter the TR receive chain through a parabolic dish antenna. The antenna has a multi-horn feed that is used to form a sum and two difference channels (Σ, ΔAZ and ΔEL).

The three signals then pass through the RF Front End (RFFE). The RFFE contains attenuators (TRLA) to protect the receiver from large signals and amplifiers (LNA) to amplify small signals. The RF Head (RFH) and Receiver subsystems then mix the signals down from carrier frequency to baseband I/Q videos. The basebanded signals are then sent to the Synchronization and Video Conversion Unit (SVCU) for further processing.

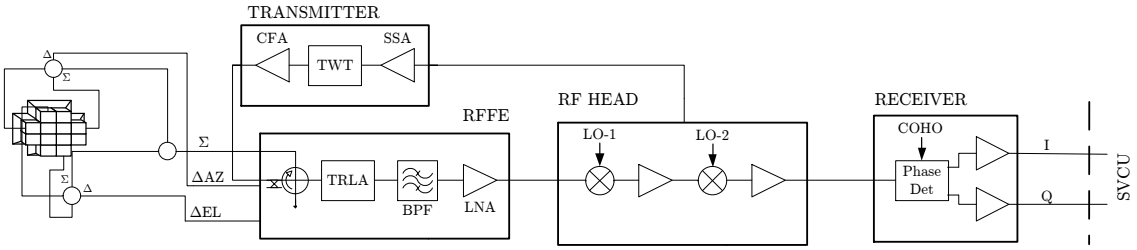


Figure 1.1: Overview of the TR transmit and receive architecture.

1.4.2 Chapter 3: Lunette System Timing Strategy and Signal Processing

The SVCU governs the timing of the radar and all its sub-systems. This chapter investigates the timing strategies the SVCU implements to fulfil the signal processing requirements of the radar system computer.

Timing

The chapter begins with a presentation of the four time scales that exist in a pulsed radar system such as the TR. The SVCU controls three of these time scales: pulse width (PW), pulse repetition interval (PRI) and the dwell time. The only time domain scale not determined by the SVCU is the wavelength, which is set by the transmit frequency (RFH).

The SVCU produces sets of L pulses for the system computer to process [9]. This is known as a radar frame, and is of length $L \times$ PRI seconds. The leading K pulses in the radar frame are filling pulses and the trailing N receiving pulses. The filling pulses ensure that, for targets at long ranges, the target echo is present in all the receiving pulses. The target video is digitized for signal processing within the receiving pulses.

Signal processing

The detection data is acquired by sampling 20 I/Q pairs of the detection video, in the N receiving PRIs, as shown in Figure 1.2 [2].

The detection data per frame is arranged into an $N \times 20$ (row$\times$col) element matrix. A 32 point FFT is then performed on each column (range gate set) of this matrix to form a range Doppler map $Y_D(k, m)$ [10]. A cell averaging CFAR detector is applied

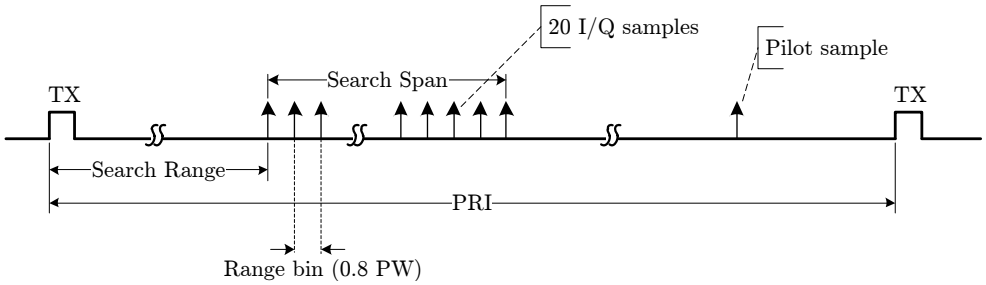


Figure 1.2: The detection channel sampling strategy.

onto this map to yield detections. The detected cells are then clustered in a nearest neighbour fashion. The centroid of each cluster (k_c, m_c) is determined, resulting in a course measurement of range and Doppler of the detected targets

$$R_{amb} = R_{search} + (m_c)(0.8 \times \text{PW})(\frac{c}{2}) \tag{1.1}$$

$$F_{d_{app}} = \frac{k_c}{32}(\text{PRF})$$

The TR tracking is done using a gating method and minimizing the difference to sum $\left(\frac{\Delta}{\Sigma}\right)$ error. The tracking video is sampled at the expected target range (centre), slightly before (early) and slightly after (late), as shown in Figure 1.3 [2].

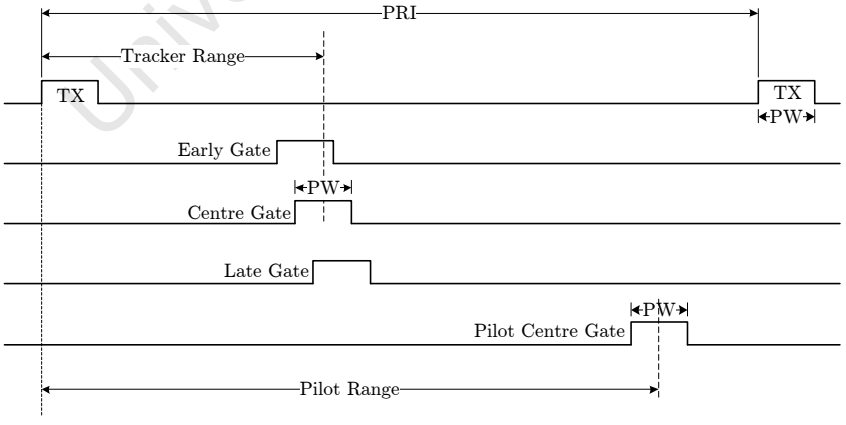


Figure 1.3: The tracking channel sampling strategy.

A DFT is then performed on the N early and late gate samples. The error in the

approximated target range is then calculated as [10]

$$\varepsilon_R = K_R^{T(T)} \text{real}\left(\frac{E-L}{E+L}\right) \tag{1.2}$$

To measure the target Doppler, a DFT is performed on the N centre gate samples at a frequency offset above and below the expected Doppler frequency. The error in the expected target Doppler is then

$$\varepsilon_D = K_D^B \text{real}\left(\frac{f_+ - f_-}{f_+ + f_-}\right) \tag{1.3}$$

The angular (azimuth and elevation) tracking errors are calculated directly from the centre gate samples of the ΣT and ΔAZ or ΔEL video channels and given by

$$\varepsilon_\theta = \text{real}\left(\frac{\Sigma}{\Delta}\frac{1}{B(l, f_c, p)}\right) \tag{1.4}$$

The tracking process within the TR strives to minimize the range, Doppler and angular errors presented above.

1.4.3 Chapter 4: Synchronization and Video Conversion Unit

Chapter four presents a comprehensive discussion of the SVCU hardware, and operation thereof.

The function of the SVCU within the TR is first discussed, followed by a general description of the cards and modules that constitute the SVCU hardware. The four primary tasks of the SVCU: (1) digitizing the receiver video signals; (2) producing sync pulses for the radar sub-systems; (3) providing time within the radar; and (4) producing composite video for the A/R scopes are then discussed in detail.

1.4.4 Chapter 5: Simulation of the Lunette Tracking Radar

This chapter describes the creation of a system level model of the tracking radar (TR). This model is used to verify the theoretical and operational understanding of the SVCU; it will also serve to produce a functional specification of the hardware required to replace the current SVCU.

The TR simulation model was primarily built in the SystemVue^{TM}(SV), with the back-end signal processing algorithms being implemented in the Python programming environment.

Each subsystem of the TR radar was built as a SV sub-network model using a combination of parts from the Algorithm design and RF design SV libraries. The subsystems were then integrated to form the TR model. The TR model was then run, at a sample rate of 10 MHz, with the following setup

Parameter	**Symbol**	**Value**
Carrier frequency	f_c	5.5 GHz
PRF	f_{PRF}	2500 Hz
Pulse width	τ	1.0 μs
Number of pulses	N	32
Detection range	R_{DETWIN}	28940 m

(a) Radar

Parameter	**Symbol**	**Value**
Range	R_{TGT}	30473 m
Doppler	f_d	684 Hz
RCS	σ	1 m^2

(b) Target

Table 1.2: The OTR tracking radar SystemVue^{TM}simulation set up.

Simulation results

Figure 1.4 shows the basebanded video (output of Receiver model) of the detection window. The 'butterfly' or 'onion' depicted in the figure is caused by the target Doppler [11]. The video is then sampled and processed by the SVCU model.

The sampled data is then fed into the vector processing unit (VPU) model to perform the FFTs. The VPU model was built using the SV *MathLang* part. MathLang enables scripts to be written in a Matlab-like language with signal processing libraries [12]. The output of the VPU was then written to binary files to be further processed in Python.

Figure 1.5 shows a plot of the range Doppler map in Python.

A simple thresholding routine reveals the target centre to be in range bin 13 and Doppler bin 9. From Equation 1.1 above, a target is detected at a range 30500 m with a Doppler frequency of 703 Hz. This is within one range-Doppler cell resolution of the actual target

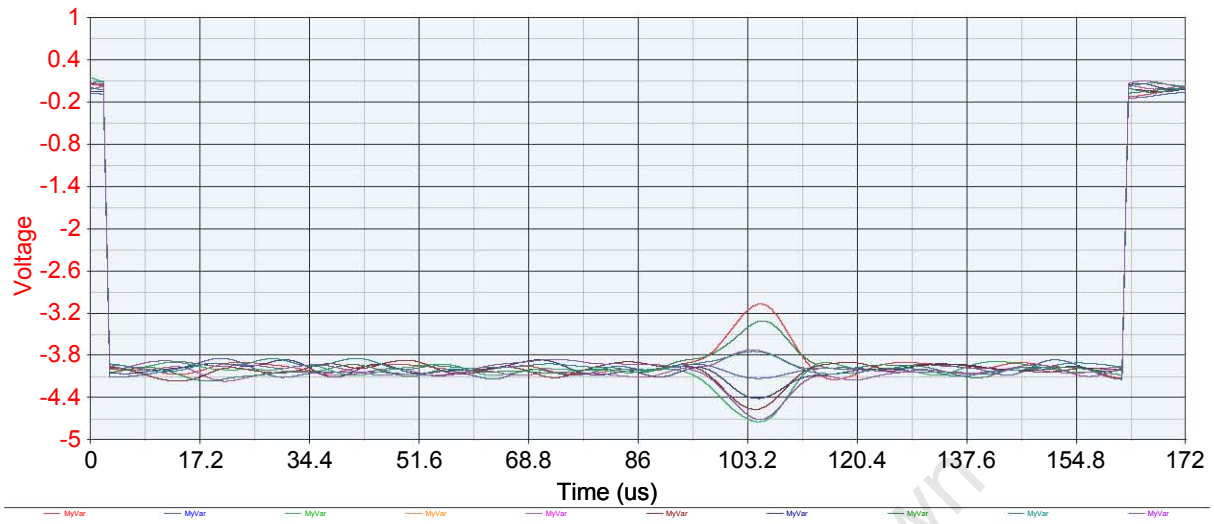


Figure 1.4: SV Plot: The detection window.

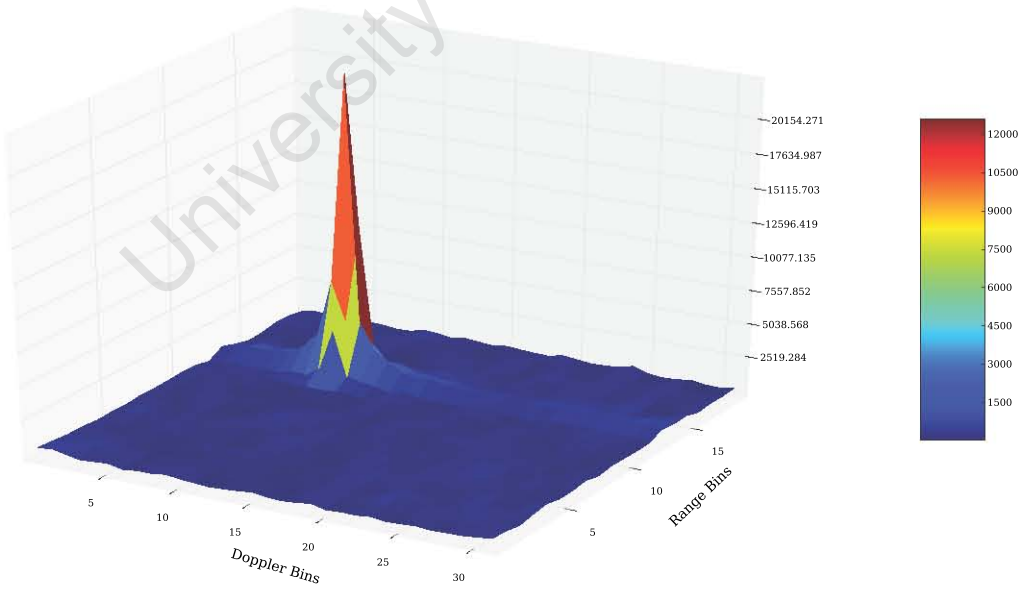


Figure 1.5: Python: The range Doppler map.

parameters (Table 1.2).

To measure the target range and Doppler to a finer resolution, the SVCU tracking channel and accompanying tracking gates are used. This forms part of the critical timing signal analysis and is investigated the next chapter.

1.4.5 Chapter 6: Characterisation of Critical Timing Signals in the SVCU

This chapter describes the measurement/tracking operation within the TR and the timing critical signals within the SVCU that drive this process.

The SystemVue^{TM}(SV) tracking channel model was run at a system sample rate (SSR) of 1 GHz. The early, centre and late tracking gates were set to sample the ΣT signal around the detected target range, R_{det}, as determined from the detection model simulation.

A DFT of the N early and late gate samples was then performed. The error between the estimated (from detection model run) and actual target range was then calculated as per Equation 1.2.

For fine Doppler measurement, a DFT of the centre gate sample was done at a frequency slightly above and slightly below the detected Doppler frequency ($f_{d_{det}}$). These two values were then used to calculate the error between the estimated and actual target Doppler frequency as per Equation 1.3.

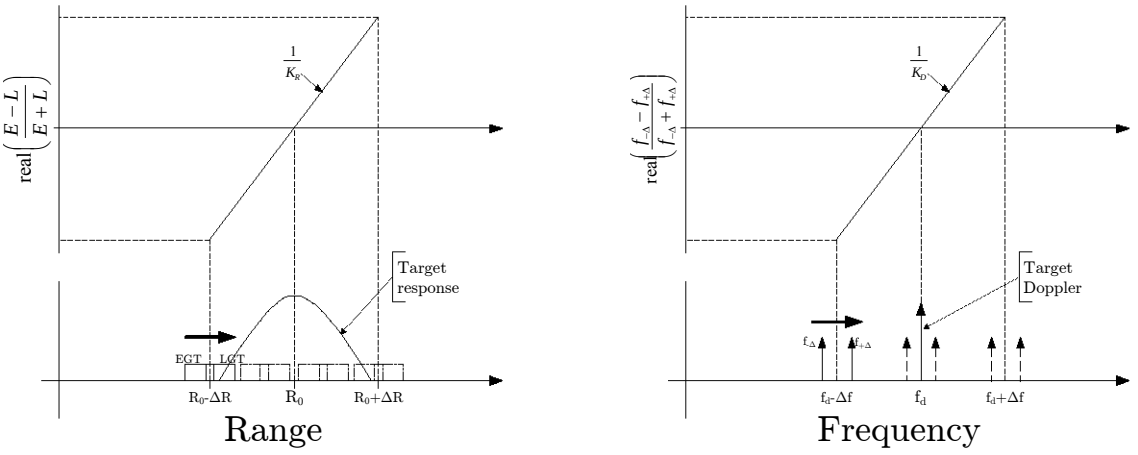
Factors $K_R^{T(T)}$ and K_D^B, used to convert the electrical signal errors to a physical range and Doppler error, were calculated by building the TR error models.

TR error models

The TR errors models were developed by sweeping the range and Doppler gates across a target echo with a known range and Doppler frequency. This calibration process is shown in Figure 1.6.

Tracking channel output

The measurement signal processing yielded a range and Doppler error of $\varepsilon_R = 19.75$ m and $\varepsilon_D = 18.96$ Hz. Combining this with the detection estimate, the TR measures the



(a) Range

(b) Dopler

Figure 1.6: Methodology used to build the TR measurement error models.

target at

$$\begin{aligned} R_{track} &= R_{det} - \varepsilon_R = 30480.24 \text{ m} \\ f_{d_{track}} &= f_{d_{det}} - \varepsilon_D = 684.04 \text{ Hz} \end{aligned}$$

This is within 1 SI unit of the actual target model parameters (Table 1.2).

Corruption of critical timing signals

Once the tracking channel model had been verified, the measurement gates were centred on a simulated target (such that $\varepsilon_R \approx 0$). The gates were then corrupted with rise time, offsets and jitter (zero-mean). In each instance a Gaussian analysis was done on a set of 100 measurements. The 3σ value was taken as a figure of merit to evaluate the error on the measurements.

Rise time and positional offsets on the measurement gates resulted in fixed errors ($3\sigma = 0$) on the TR range and Doppler measurements. This can be countered through calibration. A 10 ns jitter on the measurement gates resulted in Figure 1.7, here the 3σ range error is half a metre.

The 3σ error values for range and Doppler for increasing jitter on the measurement gates is shown in Figure 1.8.

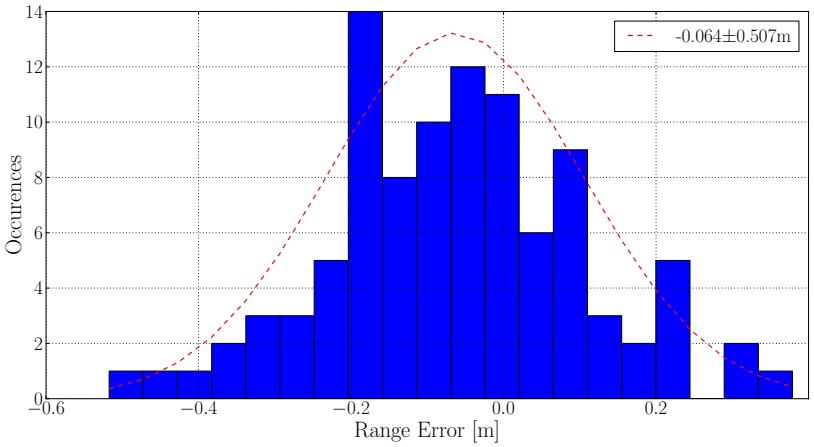


Figure 1.7: Python: Jitter introduced into measurement gates.

Proposed new SVCU

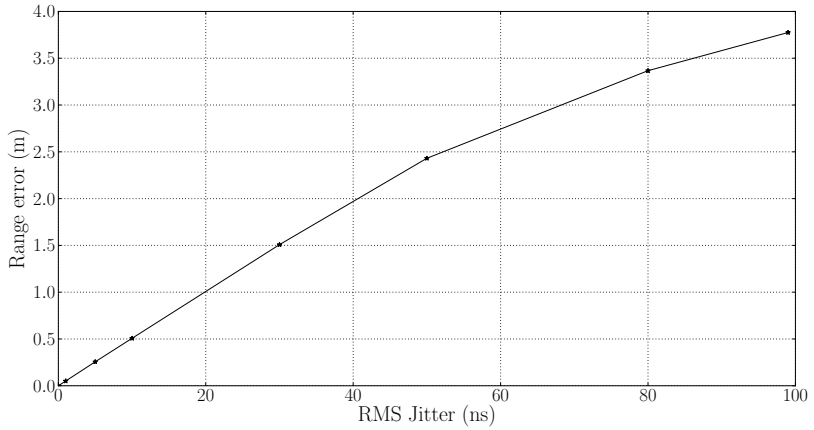
A new SVCU architecture is then proposed, using a fast ADC upfront and performing the tracking gate integration digitally. For the purpose of the simulation the Texas Instruments (TI) ADS62P49 was modelled in SV [13].

Differential and integral non-linearities were introduced (values from data sheet) and resulted in fixed range and Doppler errors. The effects of aperture jitter on the ADC were then investigated. The total aperture jitter value was calculated as the root-sum-square of the ADC aperture jitter (from data sheet [13]) and the sample clock jitter [14]. Assuming the ADC sample clock would be a derivative of the TR master oscillator (MOSC), the sample clock jitter was obtained by integrating the single-sideband phase noise characteristic of the MOSC [15].

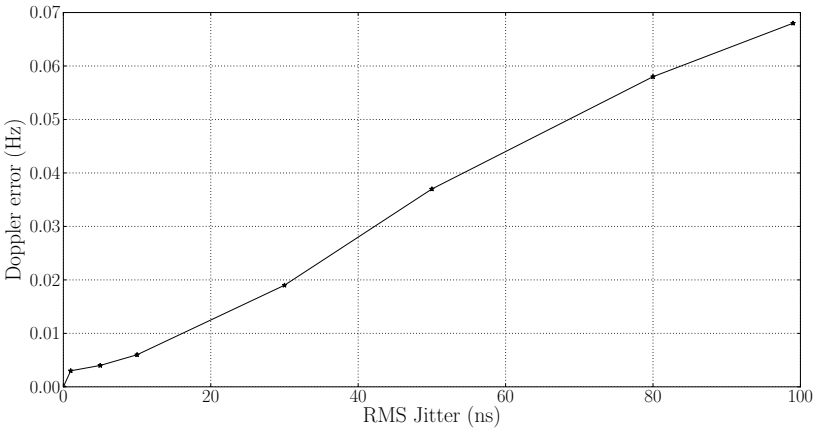
Figure 1.9 shows the effect of aperture jitter on the TR range measurement. This is orders of magnitude smaller than the effect of jitter on the tracking gates.

Jitter was then introduced onto the measurement gates that drive the digital integrators. The results are captured in Figure 1.10.

These errors are similar to those predicted for the original SVCU layout using analogue integrators (Figure 1.8).



(a) Range



(b) Doppler

Figure 1.8: Python: Effect of jitter on radar measurements.

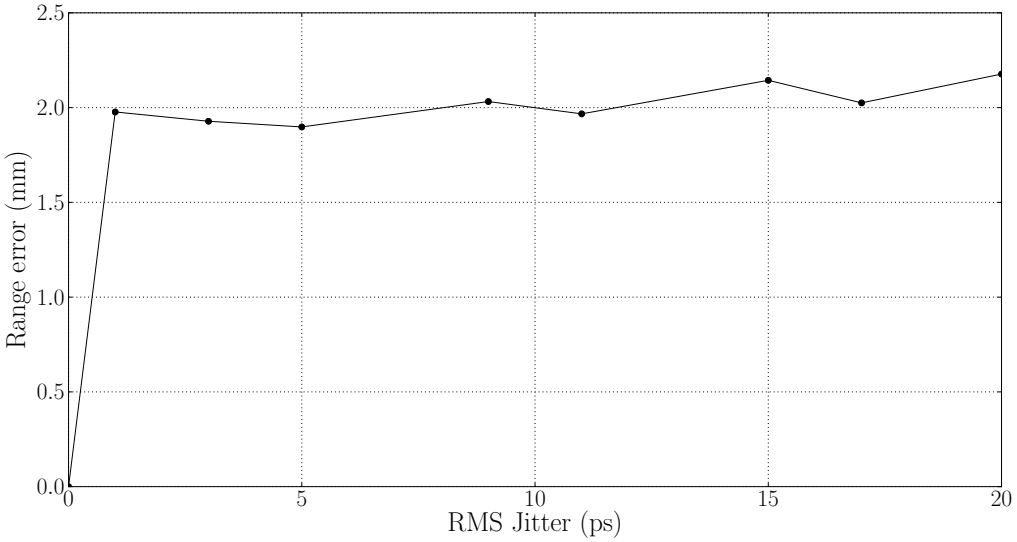


Figure 1.9: Python: Effect of aperture jitter on TR range measurement.

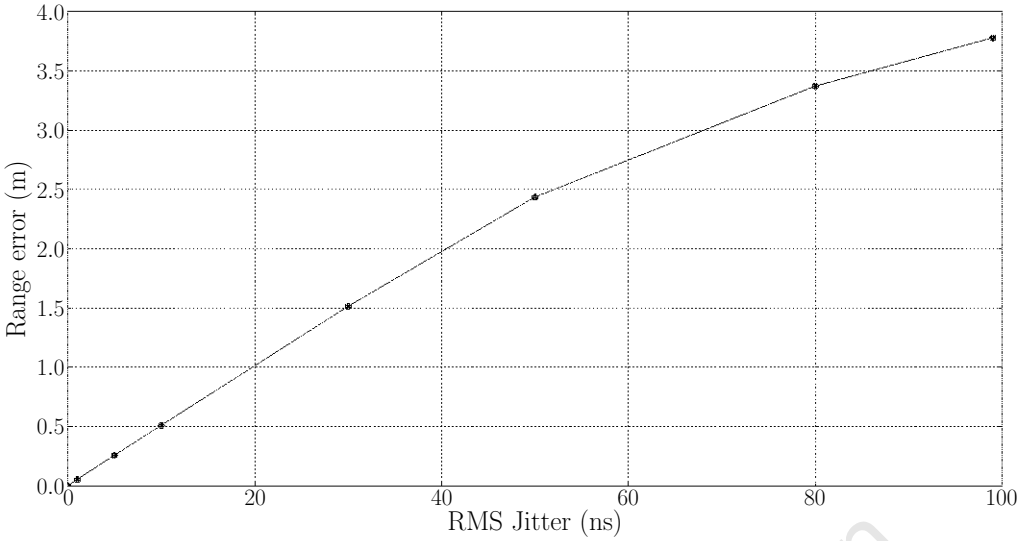


Figure 1.10: Python: Effect of jitter on digital integrators.

TR jitter budget

Given that the range accuracy deteriorates at 0.5 m per 10 ns of jitter, the TR jitter budget is set at 30 ns in order to achieve the specified range accuracy of 1.5 m *rms*.

The synchronization pulses that govern the measurement process are the TX pulse, the ADC sampling clock and the tracking/measurement gates. This yields the following constraint

$$\text{Total}_{jit} = \sqrt{\text{TX}_{jit}{}^2 + \text{Aper}_{jit}{}^2 + \text{Gate}_{jit}{}^2} \leq 30\text{ns} \tag{1.5}$$

An FPGA jitter case study

A study was then done to determine the jitter budget achievable in a Xilinx Spartan-6 FPGA based SVCU design. The study found that, through the careful use of the Spartan-6 dedicated clocking resources and a synchronous design, a total jitter of

$$\text{FPGA\_T}_{jit} \;=\; \sqrt{\text{Inter}_{jit}^2 + 36824}\ \text{ps} \tag{1.6}$$

is achievable. Here Inter_{jit} is the jitter introduced by the interface routing the TX sync

from the FPGA (SVCU) to the transmitter rack.

By equating this to the total jitter budget of 30 ns, we conclude that: in an FPGA based design, the jitter introduced by the interface sending the TX sync to the transmitter rack will, almost entirely, determine the range accuracy of the TR.

1.4.6 Chapter 7: Conclusions and Future Work

This chapter discusses the conclusions drawn, based upon the results gained from the previous chapters. It is shown that the primary objectives have been achieved by building a working, system level, model of the SVCU. To validate this model the TR receive-transmit chain and signal processing were simulated. A critical timing study of the SVCU synchronization pulses was done.

Considerations for SVCU redesign

A new SVCU architecture was simulated, using a fast ADC (TI ADS62P49) upfront and performing the tracking gate integration digitally.

A timing study on this model found that:

- Differential and integral non-linearities within the ADC model produced fixed offsets in the range and Doppler measurements
- For an ADC aperture jitter of 20 ps a range error of ± 2 mm is predicted
- From the ADS62P49 data sheet and the TR master oscillator specifications, it was established that an aperture jitter of 839 fs could be achieved
- Jitter introduced onto the tracking gates that drive the digital integrators produced a range error of ± 0.5 m per ten nanoseconds

To meet the specified TR range measurement accuracy of 1.5 m *rms* (from [1]), the total *rms* jitter on the range measurement process must not exceed 30 ns.

The synchronization pulses that govern the measurement process are the TX pulse (synchronize transmitted RF pulses), the ADC sampling clock (aperture jitter) and the tracking/measurement gates. This translates to

$$\text{Total}_{jit} = \sqrt{\text{TX}_{jit}{}^2 + \text{Aper}_{jit}{}^2 + \text{Gate}_{jit}{}^2} \leq 30\text{ns} \tag{1.7}$$

An FPGA based SVCU design

A study was then done to determine the jitter budget achievable in an FPGA based SVCU design. The Xilinx Spartan-6 FPGA was chosen for this investigation.

The study concluded that: in an FPGA based design, the jitter introduced by the interface sending the TX sync to the transmitter rack will, almost entirely, determine the range accuracy of the TR.

The circuits used to transmit and receive the TX sync are thus critical. These circuits, and the components thereof, will have to be carefully selected, tested and calibrated until the required performance (jitter, rise time and overshoot) is achieved.

Future work and recommendations

Figure 1.11 shows a block diagram of a recommended architecture for a future SVCU.

The design is centred around a COTS board with reconfigurable logic (FPGA), memory and a microprocessor. The Reconfigurable Hardware Interface for computatioN and radiO (RHINO) board developed by the RRSG group at UCT is a good example [16]. The RHINO comprises a Xilinx Spartan-6 FPGA, a TI ARM Cortex-A8 processor and also provides various peripheral functions such as USB, Ethernet and video out.

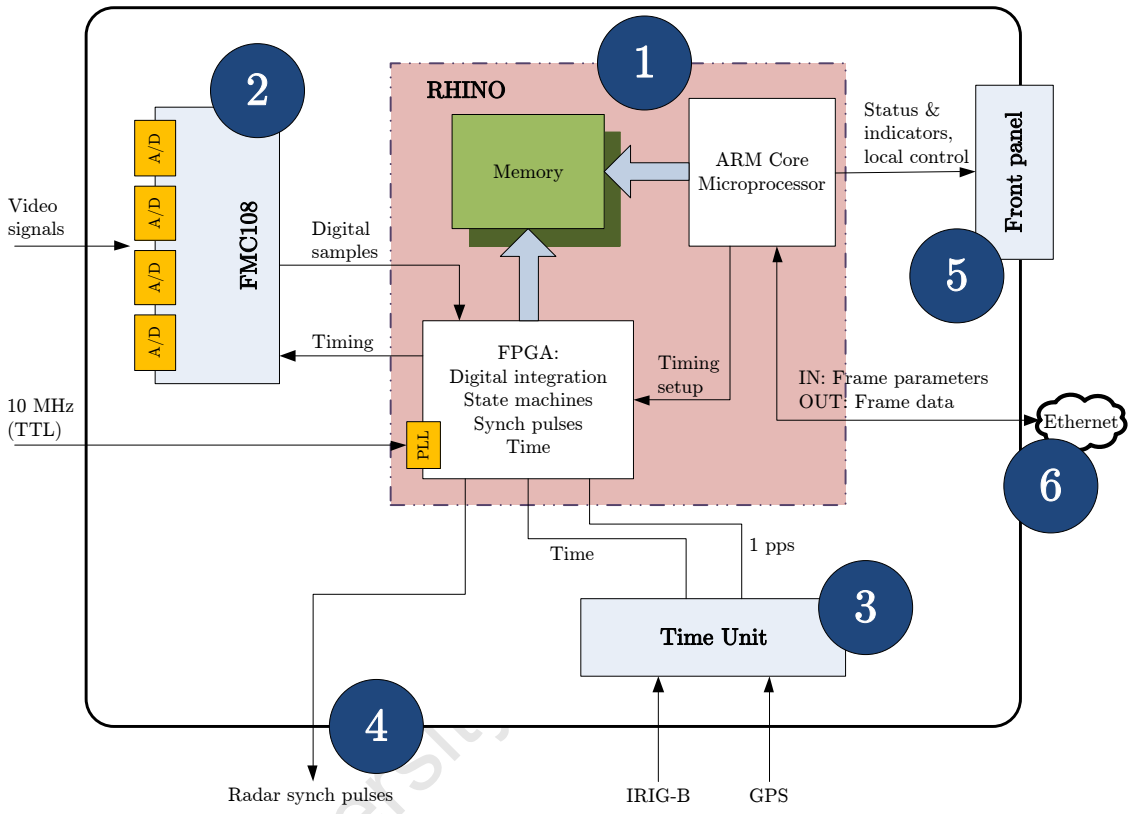


Figure 1.11: New SVCU design centred on RHINO board. At the heart of this system will be a COTS board with reconfigurable logic (FPGA), memory and a microprocessor. The analogue to digital conversion will be done by means of a FPGA Mezzanine Card (FMC) and the time of day acquired using a commercial time server. The synchronization pulses (syncs) output to the various TR subsystems will be generated within the FPGA. A front panel will show status and fault indicators and also allow local control of the SVCU. The 1553 and HSD interfaces will be replaced by a single Ethernet link. Ethernet is a ubiquitous industry standard and will allow multiple systems to communicate with the SVCU.

Chapter 2

Lunette Tracking Radar

This chapter introduces the Lunette Tracking Radar (TR) to the reader. The chapter begins with a short background of Denel Overberg Test Range (OTR). The technical specifications and an overview of the main sub-systems of the TR is then presented. The signal path that a RF target echo follows down the TR receive chain is then shown, culminating in its output to the synchronization and video conversion unit (SVCU).

The information presented in this chapter, although not part of the core functioning of the SVCU, gives the reader a good understanding of the transformations the target echo goes through before it is sampled by the SVCU. It was included for readers not familiar with precision tracking radars and also aids in the interpretation of the TR signal processing.

2.1 Denel Overberg Test Range (OTR)

Denel Overberg Test Range is a multi-purpose test range specialising in in-flight systems performance measurements for the local and international aerospace industries. The Test Range is located immediately east of the remote southernmost tip of the African continent, at latitude of 34° south, on the south-eastern coast of the Western Cape (see Figure 2.1).

The Test Range operates a comprehensive array of instruments to maintain the level of proficiency and reliability required for modern flight-testing. These include high precision radars, high-speed optical and infra-red tracking systems, fixed cameras, telemetry, meteorology and an extensive command and control infrastructure.

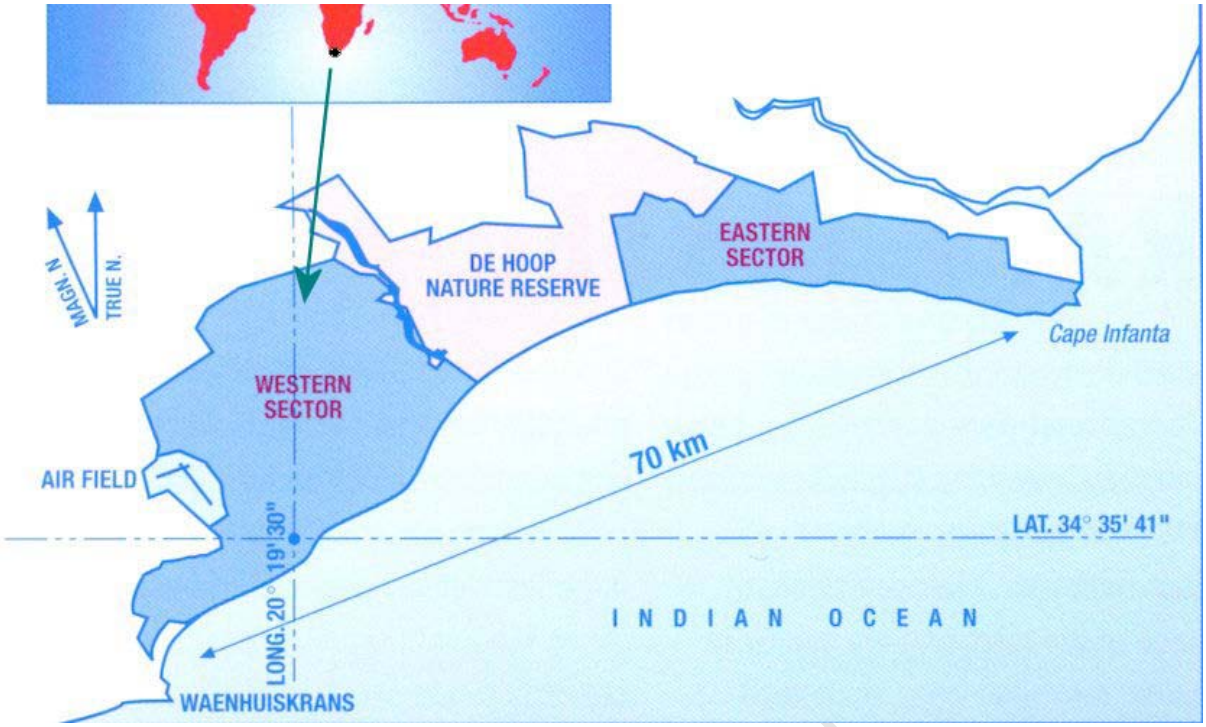


Figure 2.1: Denel Overberg Test Range (OTR)

2.2 Tracking Radar Overview

The OTR tracking radar (TR) is a pulsed Doppler, *coherent*, instrumentation radar [1]. It has the ability to track targets both in skin and beacon (transponder) mode, with a high measurement precision in range, azimuth, elevation and radial velocity.

The main system components of the TR are (Figure 2.3 shows a general block diagram of the radar) [17]:

Antenna assembly

The antenna is a 4 m (diameter) bi-axial (azimuth and elevation) type constructed of composite materials (graphite-epoxy). The antenna is in a Cassegrain configuration (reflector, sub reflector, feed) and has three main polarizations (vertical, right-hand circular and left-hand circular), controlled by the CCA. See [18] for a detailed description of the antenna system.

The feed of the antenna is a multi horn type and is used to form sum and difference channels. This is discussed in greater detail in Section 2.3.2.

Figure 2.2: Lunette Tracking Radar (TR)

Pedestal assembly

The pedestal is an "elevation over azimuth" type and houses the antenna, RF Front End (RFFE) and optical systems [1]. Accurate sensors, in the form of shaft encoders and tilt sensors, enable the measurement of pedestal direction and tilt in relation to the earth.

In addition, the pedestal includes elevation and azimuth rotary joints and azimuth slip rings. These facilitate the transfer of RF signals (receiver rack) and commands and indications (system computer) to and from equipment located on the pedestal.

Electro-optical system

The electro-optical system consists of two TV cameras. These are mounted on the pedestal, parallel to the antenna boresight, "looking through" small apertures in the antenna (see Figure 2.2).

The first (primary) camera has a long focal length telescope (120″) with a high measurement accuracy. This is connected to a TV tracker and is used during calibrations (star) for pedestal and time parameter calibrations.

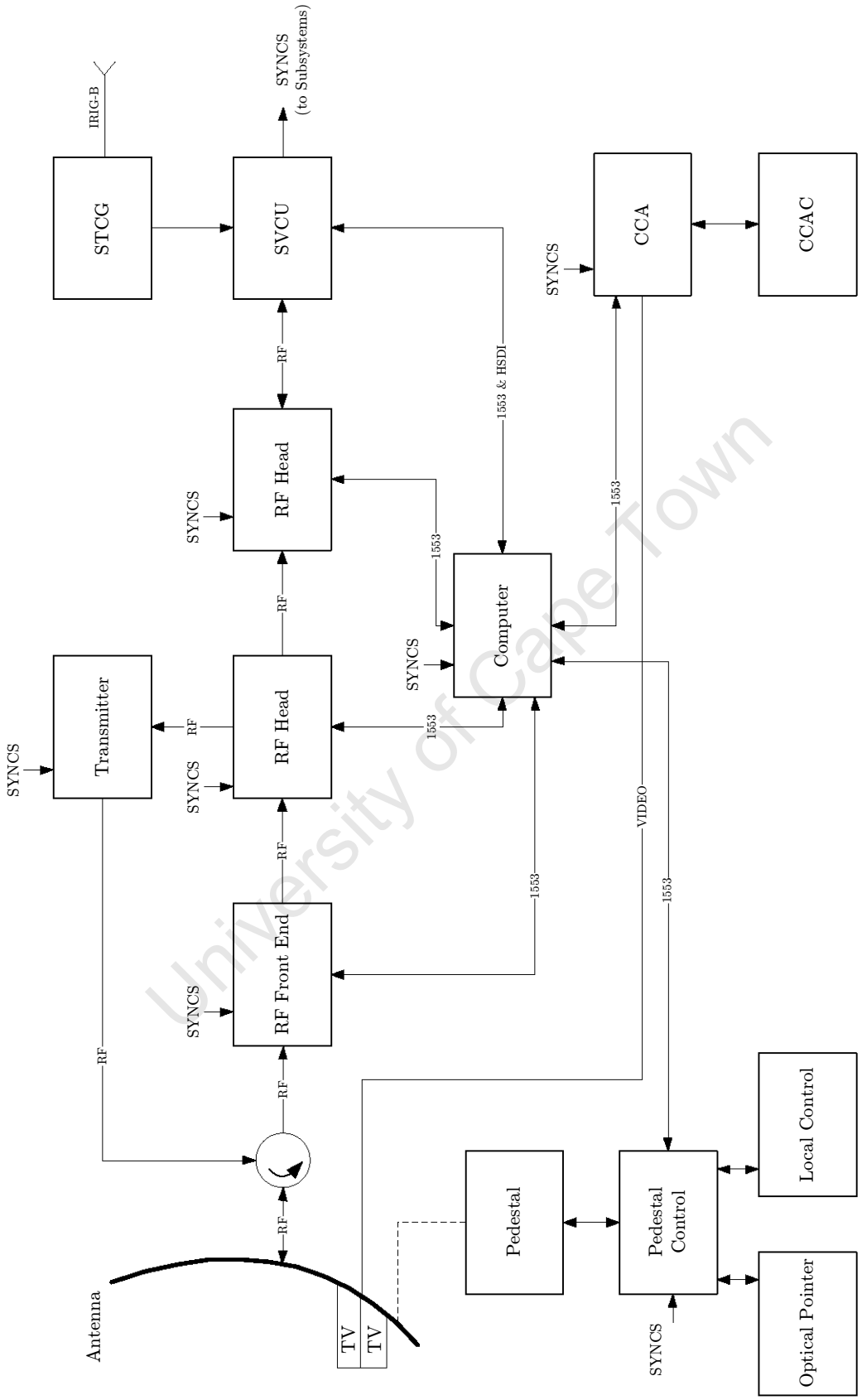


Figure 2.3: A general block diagram of the OTR tracking radar.

The secondary camera has a wide angle zoom lens and is used for general orientation and target acquisition.

Transmitter chain

The transmitter receives C-band RF pulses from the RF Head (RFH). The transmitter amplifies these pulses to reach a peak power of 600 kW at an average power value of 1.25 kW. The transmitter chain is discussed in Section 2.3.1.

Receiving system

The receiving system receives the echo signal reflected from the target in the form of three (Σ, ΔAZ and ΔEL) near identical channels. It converts these RF signals (carrier frequency) down to baseband video. In order to do this the receiving system contains stable and precise frequency sources and facilities for gain and phase control.

The receiving system is comprised of three sub-assemblies and is described in Section 2.3.2.

Synchronization and Video Conversion Unit (SVCU)

The SVCU sits at the end of the receiving chain and serves as the analogue to digital interface. Along with this task the SVCU also synchronizes the entire radar by providing clock pulses to all the TR sub-units.

The SVCU is the focus of this dissertation, thus a comprehensive discussion of its theory is presented in chapter 4.

Data processing and signal processing computer system

The computer system, within the TR, functions both as a system controller and as a signal processor. The system is fed with the digital data from the SVCU (via HSD interface), and status information from the various sub-assemblies (via 1553 MUXBUS).

Using this information the computer system performs real time (RT) target detection and tracking, information recording and dialogue with the TR subsystems.

The core principles of the TR signal processing, detection and tracking methodologies, are discussed in chapter 3.

Command and Control Assembly (CCA)

The CCA constitutes the interface between the operator and the TR, it is the operators' station. The console includes display and operation facilities to enable the operator to search and acquire targets, perform calibrations and run service or maintenance routines.

The primary actuators and displays the radar operator uses perform these tasks are:

- Joysticks (angle and range)
- Push buttons
- TTY Terminal (computer messages and commands)
- A/R scopes (skin and beacon)
- Range-Doppler display
- Optics screens (TVs - narrow angle and wide angle camera)

2.2.1 Operational and Technical Data

Here the main performance and technical parameters, as documented in [1], are summarised.

Operational performance data

Coverage and dynamic target properties:

Maximum range : 180 km (skin mode, 1 m^2 target RCS)
: 2800 km (beacon mode)
Maximum velocity : 10 km/sec (radial)
: 42°/sec (angular)
Maximum acceleration : 300 m/sec^2 (radial)
: 20°/sec^2 (angular)

System accuracies for targets echoes received at SNR = 20 dB:

Table 2.1: TR system accuracies

Parameter	**Bias**	**Random (rms)**	**Off-Boresight**
Range	1 m	1.5 m	3 m
Angles	100 μrad	100 μrad	100 mrad
Radial velocity	3 cm/sec	5 cm/sec	10 cm/sec

Technical parameters

Frequency	: 5400 - 5900 MHz (in 5 MHz steps)
Peak power	: 600 kW (TWT + CFA)
Pulse width	: 0.5, 1.0 or 2.5 μs
PRF	: 300 - 3000 Hz
Antenna gain	: 43 dB
Antenna beam width	: 1°

2.3 RF Signal Path Within the TR

The fundamental principle of radar is based on transmitting an RF signal, which reflects off a target (ship, aircraft, land or sea), and detecting the corresponding echo that returns to the radar [3]. The time taken from transmission of the RF signal to reception of its echo, T_R, is measured. Since electromagnetic signals travel at the speed of light ($c \approx 3 \times 10^8$ m/s), the distance, or range R , to the target is given by:

$$R = \frac{cT_R}{2} \tag{2.1}$$

In the case of the TR, the transmitted waveform is a pulse-modulated (simple pulse) sinusoidal wave at a C-band carrier frequency f_c [17].

This section describes how the transmit pulse is generated and amplified, the formation of the sum and difference receive channels, and the down-conversion of these channels from carrier frequency to baseband bipolar video.

2.3.1 TR Transmit Chain

The transmitter receives C-band RF pulses from the RF Head (RFH) at a nominal level of 0.025 W [8]. The transmitter amplifies these pulses to reach a peak power of 600 kW at an average power value of 1.25 kW (determined by duty cyle $\rightarrow \frac{\text{PW}}{\text{PRI}}$). This is achieved through a cascade of three amplification stages:

1. Solid state amplifier (SSA) 1 W peak output power;
2. Travelling Wave Tube (TWT) 50 - 60 kW peak output power;
3. Cross Field Amplifier (CFA) 450 - 600 kW peak output power.

The transmit pulses are synchronized to the TX pulses from the SVCU. This pulsed amplifier design ensures a *coherent* transmitter [11].

2.3.2 TR Receive Chain

Here the RF path a signal follows, from carrier frequency f_c down to baseband (received echo) and from baseband to f_c (transmit/pilot pulse), in the receive chain is discussed.

Antenna

The TR antenna is used for both transmission and reception of RF signals [1]. This is enabled by the use of a circulator and receive protector (TRLA) in the RF Front End (RFFE) [3].

Reception All RF signals arriving at the antenna are focussed, by the parabolic dish and sub reflector geometry, into the antenna feed [18]. The TR has a multi-horn feed with 12 horns. The outputs of the 12 horns are passed through a monopulse comparator to form a sum and two difference channels (Σ, ΔAZ and ΔEL).

The channels are formed, in the "plumbing" or waveguide network, by adding the various outputs of the horns whilst keeping the signal amplitude constant but changing the phase as required. This is illustrated in Figure 2.4.

Transmission RF transmit pulses arrive at the antenna on the Σ channel of the feed horn [18]. This signal is then focussed into a conical beam by the subreflector-parabolic dish geometry.

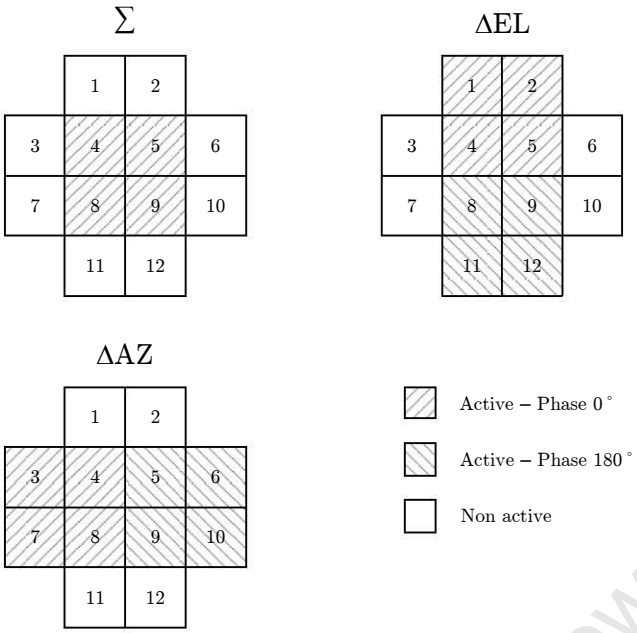


Figure 2.4: Σ and Δ channel formation in the TR 12-horn feed.

RF Front End (RFFE)

The RFFE is located on the pedestal and is the first of three sub-systems that form the receiving system. It consists of three identical (bar circulator) channels comprising of [7]:

1. A circulator (Σ channel only) to allow the antenna to be used both to transmit pulses, and receive the echoes. Although the circulator does provide isolation between the transmitter and receiver ports, it is seldom enough to protect the receiver [3]. This requires a receiver protector in the form of a TRLA (see below).
2. Bandpass filter to filter out signals outside the signal bandwidth B (5.4 - 5.9 GHz).
3. Transmit Receive Limiter Attenuator (TRLA) to protect the receiver from any TX leakage and to attenuate large received signals.
4. Low Noise Amplifier (LNA) to amplify small receive signals.

RF Head (RFH)

The RFH generates all the frequencies necessary in the radar, to down-convert received signals and up-convert transmit and pilot pulses, synchronised to a 120 MHz master

crystal oscillator (MOSC). Synthesizing all frequencies within the radar from a single stable local oscillator, MOSC, ensures coherency in the receive chain [5].[1]

Received signal down-conversion The RFH converts the three signal channels received from the RFFE down to an intermediate frequency (IF) [7]. A two stage heterodyne method is used to do this. The local oscillators (LO) required to mix down the signals are synthesized from the 120 MHz MOSC. Refer to Figure 2.5.

The sum channel is split before the down-conversions. Its new branch, ΣV, is used for display purposes (A/R scope).

The first local oscillator (LO-1) converts the signals from carrier frequency to an IF1 of 1080 MHz. The source of this LO can either be a C-band synthesizer, SYNTH-C referenced to the MOSC, or from a separate voltage controlled oscillator (VCO) depending on the mode of operation (skin or beacon).

The second LO (LO-2) converts the 1080 MHz signal down to 30 MHz (IF2). The conversions of the signals are done in parallel, as shown in Figure 2.5, and the resulting signals are sent to the receiver.

Generation of TX drive and pilot pulses The MOSC and LOs, used in down-converting the received signals, are also used to generate and up-convert the transmit drive and pilot pulses.

The TX drive pulse is formed by pulsing the CW 120 MHz MOSC using the PTX pulse from the SVCU. This 120 MHz pulse is multiplied, ×9 frequency multiplier, up to 1080 MHz and then mixed with the SYNTH-C to bring it to f_c. This 5 - 17 μs* pulse is fed to the transmitter as the TX drive.

The pilot pulse is generated in a similar manner to the TX drive pulse. A CW 30 MHz (synthesized from MOSC) signal is pulsed by the IPLT sync from the SVCU. This pulse is then up-converted to either 1080 MHz (BITE-1) or f_c = 5.4 - 5.9 GHz (BITE-2). This allows the pilot to be injected in the RFFE or after LO-1 in the RFH.

Receiver

The receiver down-converts the signals from IF2 to baseband video signals [7]. This is done by means of a phase detector, referred to in radar literature as a *coherent* or I/Q

[1] In beacon mode the LO-1 frequency in generated by a VCO. Beacon mode is thus non-coherent

*This is independent of the radar pulse width. The pulse width is set in the transmitter and synchronized using the TX sync from the SVCU

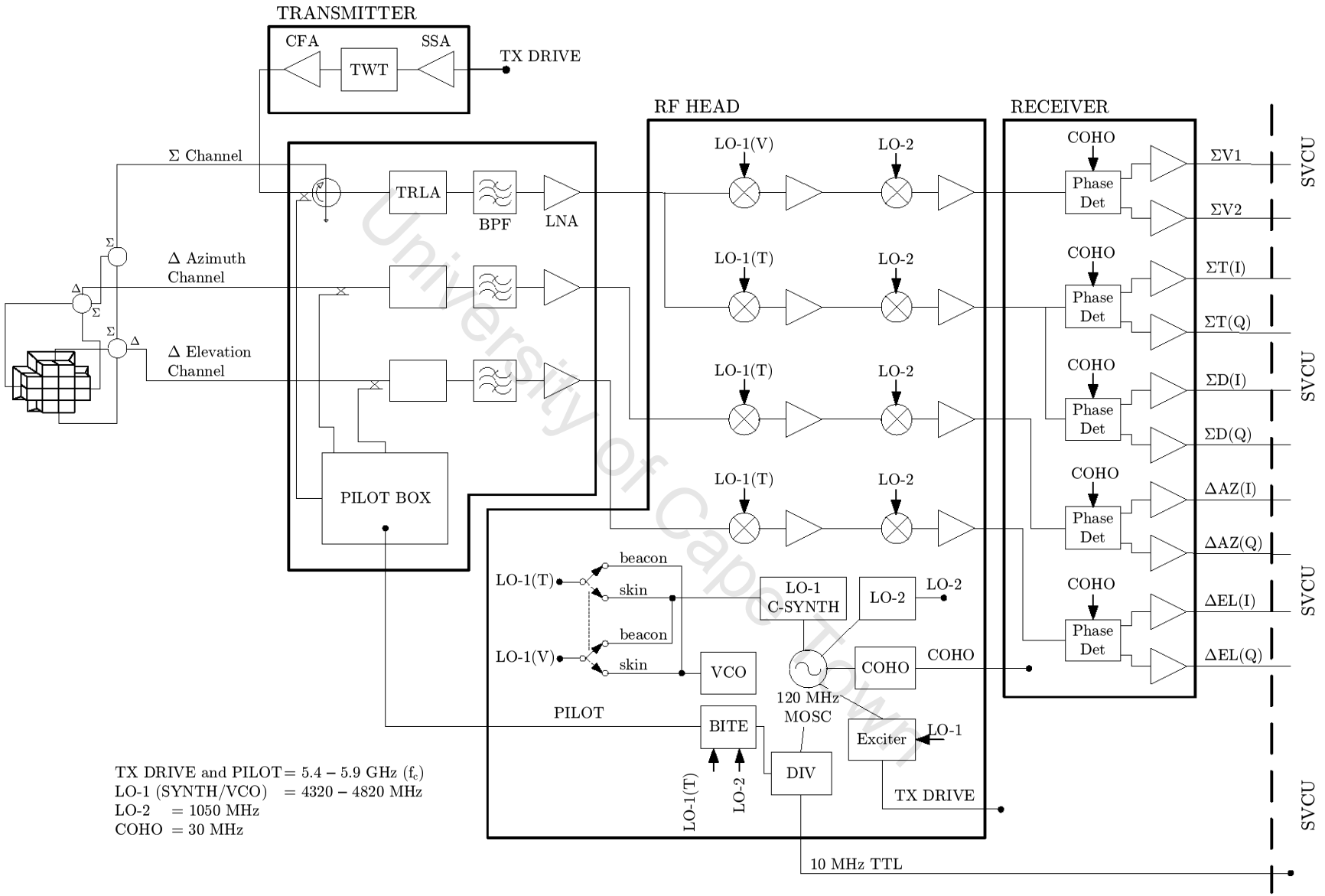


Figure 2.5: The tracking radar transmit and receive RF chain.

detector. The *coherent* oscillator, required for the detector, is a derivative ($\div 4$) of the 120 MHz MOSC and is received from the RF Head.

Before the down-conversion the sum channel is again split into two channels, ΣD for detection and ΣT for tracking. The five channels (ΣD, ΣT, ΣV, ΔAZ, ΔEL) are then down-converted, using phase detectors, resulting in bipolar video signals.

These signals are comprised of an in-phase and a quadrature channel, labelled 'I' and 'Q'[†].

The ten signals are then output to the SVCU for further processing.

2.3.3 Frequency Analysis of RF Signal Path

Figure 2.6 shows the two stage superheterodyne architecture of the TR receive chain. For a transmitted signal $Ax_p(t)\cos(2\pi f_c t + \theta_{OSC})$, the received signal is modelled as [3] [5]

$$r(t) = A_1 x_p(t')\cos(2\pi(f_c + f_d)t' + \theta(R)) \tag{2.2}$$

where A_1 is the attenuation factor due to atmospheric and other losses, $x_p(t)$ is the transmitted finite pulse train, $t' = t - t_R$ is the time delay due to the target range and $\theta(R)$ the phase shift of the returned signal due to the target range and the arbitrary initial phase of the TR MOSC [5] [2]

$$\theta(R) = \theta_{OSC} - \frac{4\pi}{\lambda}R \tag{2.3}$$

This received signal is down-converted in frequency, in a two stage process, and brought to baseband using a *coherent* I/Q phase detector. This results in the in-phase and quadrature phase videos

[†]The ΣV channel is labelled as ΣV1 and ΣV2 as it is only used for display purposes.

[2]For a stationary target $f_d = 0$ and $\theta(R) = K$, a constant number that is a function of the range to the target [11].

$$
\begin{aligned}
I(t) &= A_2 x_p(t') \cos(2\pi f_d t' + \theta(R)) \\
Q(t) &= A_3 x_p(t') \sin(2\pi f_d t' + \theta(R))
\end{aligned} \tag{2.4}
$$

2.4 Summary

A short background of Denel Overberg Test Range (OTR) was given and an overview of the Lunette Tracking Radar (TR) presented. The transmitter chain of the TR was shown to be of a pulsed amplifier design consisting of three stages. The receive chain architecture consists of a two stage heterodyne mixer coupled with a phase detector. The received signal is split at various stages of the receive chain, culminating in five basebanded I/Q video pairs that are fed to the SVCU.

The next chapter discusses how these I/Q videos are processed, more specifically the timing strategy and signal processing aspects of the TR. This forms the basis of the various tasks that the SVCU is required to perform.

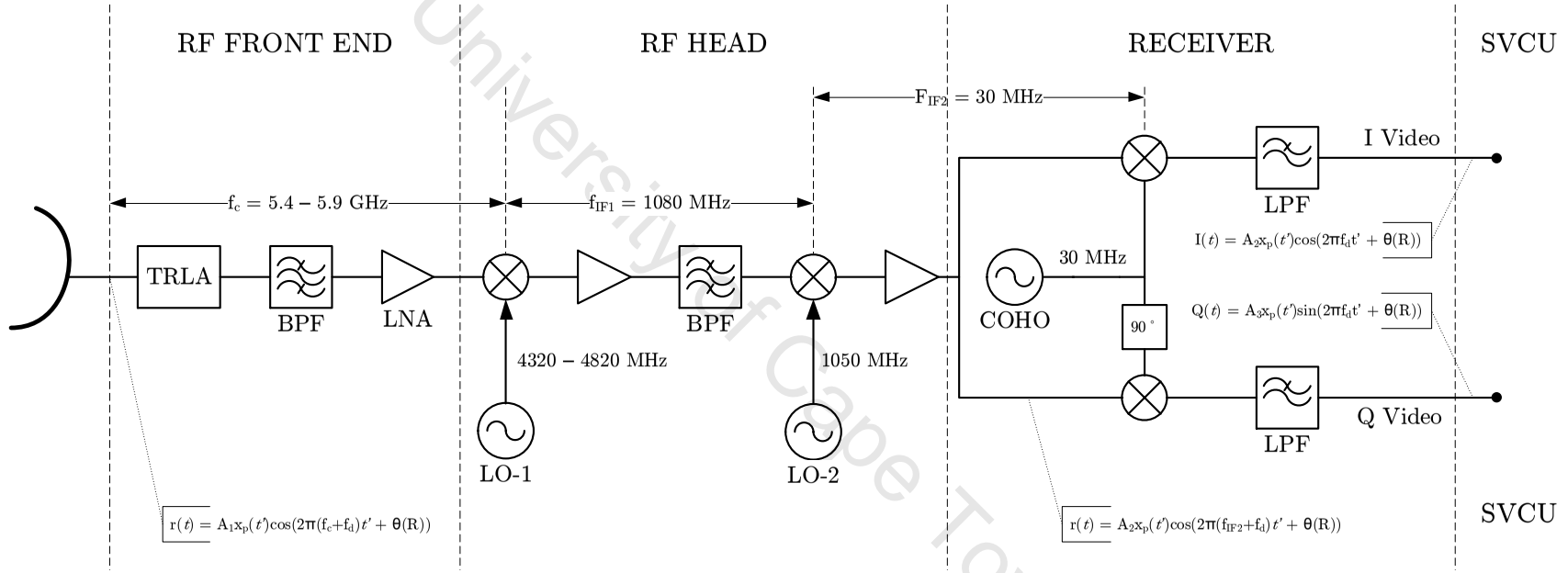


Figure 2.6: TR RF path frequency diagram.

Chapter 3

Lunette System Timing Strategy and Signal Processing

The SVCU governs the timing of the radar and all its sub-systems. This chapter investigates the timing strategies the SVCU implements to fulfil the signal processing requirements of the radar system computer.

The chapter begins with a presentation of the four time scales that exist in a pulsed radar system such as the TR. The concept of a radar frame is then shown, with descriptions of the synchronization pulses (syncs), per PRI and per frame, that make up the radar frame. The different timing modes of the TR are then mentioned, with their effects on the signal processing (SP) cycle. The chapter is concluded with an investigation of the sampling strategies, and subsequent target signal processing, of the detection and tracking channels within the radar.

3.1 Lunette Tracking Radar Timing Fundamentals

The Lunette Tracking Radar (TR) is a pulsed radar. This means the radar transmits for a very short time and then "listens"[1], for a period of time, for echoes of this transmitted signal [5]. The TR transmits a group or burst of these pulses, at a time, known as a radar frame.

[1]Listen, in this context, implies digitizing and processing the received signals to identify any possible targets.

3.1.1 The Four Time Scales of Pulsed Radar

Time domain pulsed radar signals can be described in terms of four time scales [11]. These four time scales within the TR are discussed below.

Wave length

The radar transmits its pulses at a carrier frequency f_c [4]. Given that EM waves travel at the speed of light $c \approx 3 \times 10^8$, the wave length λ is defined as

$$\lambda = \frac{c}{f_c} \tag{3.1}$$

In the case of the TR the transmit frequency can be selected between 5.4 and 5.9 GHz in steps of 5 MHz [8]. From Equation 3.1 this results in a wavelength between 50 - 55 mm.

Pulse width

The duration for which the transmitter transmits is defined as the pulse width (PW) τ. In the TR this value can be set to 0.5, 1.0 or 2.5 μs.

The transmit signal (high power) is routed to the antenna by injecting it, through a circulator, into the Σ channel [7]. For this reason, when the radar is transmitting (duration of PW), the sensitive receiver must be isolated from the antenna. This isolation is achieved using the inherent isolation of the four port (terminated in a matched load) circulator and a TRLA [3].

Pulse repetition interval

The time between two consecutive transmit pulses in a radar frame is known as the pulse repetition interval (PRI) [4]. A more common representation of the PRI is as a measure per unit of time, usually per second, and called the pulse repetition frequency (PRF). The PRF and PRI are related according to

$$\text{PRF} = \frac{1}{\text{PRI}} \tag{3.2}$$

The PRF is often expressed in Hertz and in the case of the TR is selectable between 300 and 3000 Hz [1]. The determination of the PRF is governed by the following constraints

Average transmitted power Given the peak transmit power of P_t, the average power output P_{avg} of the transmitter is expressed as [5]

$$P_{avg} = P_t \times \tau \times \text{PRF} \tag{3.3}$$

The TR transmitter has a peak RF output of 600 kW [8]. Following from Equation 3.3, the pulse width and PRF are chosen such that the average transmit power does not exceed 1.25 kW.

Radar frame length The radar frame length, RF_l measured in seconds, is calculated as [9]

$$RF_l = L \times \text{PRI} = \frac{L}{\text{PRF}} \tag{3.4}$$

where L is the number of pulses (PRIs) in the radar frame.

For target processing, within the TR, a minimum of 17 (16 *active* and 1 filling, see Section 3.2.1) pulses, and a frame length of at least 15 ms is required [10]. The frame length is also limited to 20 ms so as not to exceed the memory capacity of the signal processing computer.

The PRF is chosen such that the frame length meets these timing requirements, whilst ensuring there are at least 16 *active* pulses in the frame.

Range and Doppler ambiguities The transmission of a finite pulse train waveform, at a constant PRF, can lead to ambiguities in target range and radial velocity (Doppler) measurements [3].

Range ambiguities result from the fact that it is unknown if the echo received was from the last transmitted pulse, or from one of the pulses before. This implies: for an echo received at time T_R, the ambiguous range R_{amb} to the target is

$$R_{amb} = \frac{c(T_R + n \times \text{PRI})}{2} \quad n \in \mathbb{N}^0 \tag{3.5}$$

The Doppler shift is a change in the frequency of the received RF signal due to the motion of the target [4]. This shift in the frequency is given by

$$f_d = \frac{2v_r}{\lambda} \tag{3.6}$$

where v_r is the radial velocity of the target toward the radar and λ the wavelength.

In a pulsed radar, such as the TR, the Doppler is sampled at the PRF[‡]; thus, by Nyquist sampling theorem, the maximum Doppler frequency that can be unambiguously measured is [5]

$$f_{d_{max}} = \pm\frac{\text{PRF}}{2} \tag{3.7}$$

Any targets with a Doppler shift $|f_d| > \dfrac{\text{PRF}}{2}$ will have an apparent Doppler frequency of [5]

$$f_{d_{app}} = f_d - k \times \text{PRF} \quad k \in \mathbb{N} \tag{3.8}$$

that lies within the $\pm\dfrac{\text{PRF}}{2}$ frequency range.

During the acquisition mode of the TR, the radar switches between several PRFs (two in *non-coherent* mode and three in *coherent* mode) in order to solve for range and Doppler ambiguities [1]. During tracking the PRF is kept more or less constant.

The TR 'optimal' PRF is the highest PRF, under the constraints discussed above, such that the target is located in the middle of the PRI in range and in the middle of the Doppler filter.

[‡]This is known as the spatial Doppler [5].

Dwell time

The fourth time scale in a pulse Doppler radar is the dwell time. The dwell time is the time taken to transmit and receive the $N^{\dagger}$ pulses used for target detection [5]. These pulses are also referred to as the *active* pulses. The dwell time is given by

$$T_d = N \times \text{PRI} \tag{3.9}$$

In a radar using *coherent* processing, this time is known as the *coherent* processing interval (CPI) [5].

3.1.2 Coherency

Coherency in a radar is determined by its ability to preserve and extract the phase shift, due to the motion of a target, from the received echo signal [4]. This phase information can then be used to calculate the velocity of the target.

The TR uses a stable local oscillator, MOSC, to derive all the frequencies in the radar [7]. The use of a single frequency source for transmission and reception signals[2], coupled with an I/Q detector to bring the echo signals to baseband, ensures that the TR is *coherent*.

Although the radar receiver downconverts the analogue signals *coherently*, the SP may choose to ignore the phase information and operate in *non-coherent* skin mode [10]. One reason for this is explained below.

Target acquisition in the TR

In *coherent* mode the TR resolves both the range and velocity ambiguities of the target [1].

Multiple PRFs The TR switches between PRFs, two in *non-coherent* mode and three in *coherent* mode, in order to solve for range and Doppler ambiguities [1]. A change in the PRF requires a new radar frame. *Coherent* mode thus has a three cycle acquisition

[†]It must be noted that N is less than the L pulses transmitted per TR frame. This is because L also encompasses filling pulses, see Section 3.2.1.

[2]This is only true for skin mode. In beacon mode a separate VCO is used to down-convert the received echoes. Beacon mode is thus, as shown in Section 3.3, *non-coherent*.

delay, as compared to two cycles in *non-coherent* mode.

Doppler analysis Resolving the velocity requires a frequency spectral analysis of the received signal, since the target velocity is related to its Doppler frequency [11]. In the TR the extraction of the Doppler frequency is done by means of Fourier transforms (FFT and DFT) [10].

Due to the computational intensity of the Fourier transforms[‡] (which have to be performed within the SP-cycle) and the added delay in *coherent* mode acquisition, the TR is designed to acquire targets *non-coherently* regardless of the mode (skin or beacon) the radar is operating in [10]. Once the target is locked in range, the SP may switch to *coherent* mode if required.

3.2 The Tracking Radar Frame

A radar frame is a group of pulses transmitted at a constant PRF to ensure a meaningful radar response [9]. The complete frame, thus far referred to as TR frame or radar frame, is called the transmit frame; whereas the trailing part, where sampling and processing is performed, is known as the receive frame (refer to Figure 3.1).

3.2.1 SVCU Transmit Frame

The transmit frame is initiated at a predetermined and precisely measured absolute time of day with a resolution 1 μs (see t_0 trigger in Section 4.5.1). Transmit frames are pre-programmed to any length $17 \leq L \leq 64$ PRIs, provided the transmit frame fully overlaps the receive frame (see below).

Figure 3.1 shows that the transmit frame consists of a pre-frame, filling frame (K PRIs) and receiving frame (N PRIs). Given that the duration of the pre-frame $\ll$ PRI, the length of the transmit frame is inferred as $L = K + N$ PRIs.

[‡]The Fourier transforms are performed outside the SEL, by an accompanying Vector Processing Unit (VPU) [19]. The VPU performs repetitive, floating point, numerical computations at a very high speed. This reduces the load on the SEL.

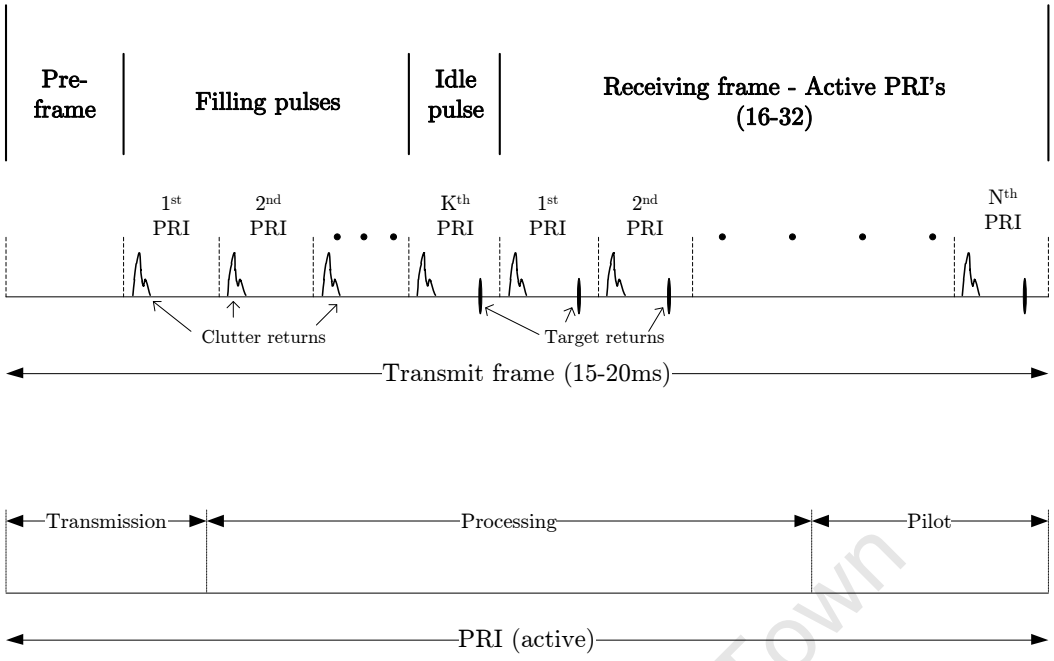


Figure 3.1: The TR transmit frame and PRI regions.

Pre-frame

This part of the frame consists of timing signals generated only once per frame, at the beginning of the frame (e.g. t_0 trigger) [20].

Filling frame

The filling frame consists of constant timing signals generated for $1 \leq K \leq 32$ PRIs. The value of K is passed to the SVCU in the command message from the SEL [21]. The filling PRIs serve three purposes: (1) to ensure the transmit frame meets the minimum frame length of 15 ms; (2) to evaluate the signal strength (idle pulse); and (3) so that all the PRIs of the receiving frame are *active* (contain echoes from the target). The last of these is discussed below.

If a target is at a range greater than the unambiguous range ($R_{ua} = \frac{c}{2\text{PRF}}$), then the echo from a transmitted pulse is only received after the nth pulse has already been transmitted [4]. Thus it is necessary to transmit extra (filling) pulses before the first receive frame pulses is processed. This is to insure that all the pulses within the receive frame are *active* and also contain the clutter echoes.

When the TR is producing consecutive transmit frames (continuation frames, see Section 3.3.1), the trailing pulses of the previous frame act as n^{th} time turnaround pulses for the current frame. In this scenario filling pulses are not needed. Having stated that, at least one filling pulse is required per transmit frame. This is called the idle pulse and is used to detect saturation in the RF receiver channels.

Receiving frame

The receiving frame consists of constant and variable timing signals generated for $16 \leq N \leq 32$ PRIs. The value of N is also part of the command message from the SEL. In the context of radar theory, from Equation 3.9, the duration of the receive frame is known as the dwell time or CPI in *coherent* mode.

Each PRI in the receiving frame, all of which are *active*, can be further broken into (Figure 3.1)

Transmission region Constant signals related to the transmission.

Processing region Consists of variable signals used for detection (search) and tracking (measurement) sampling and processing.

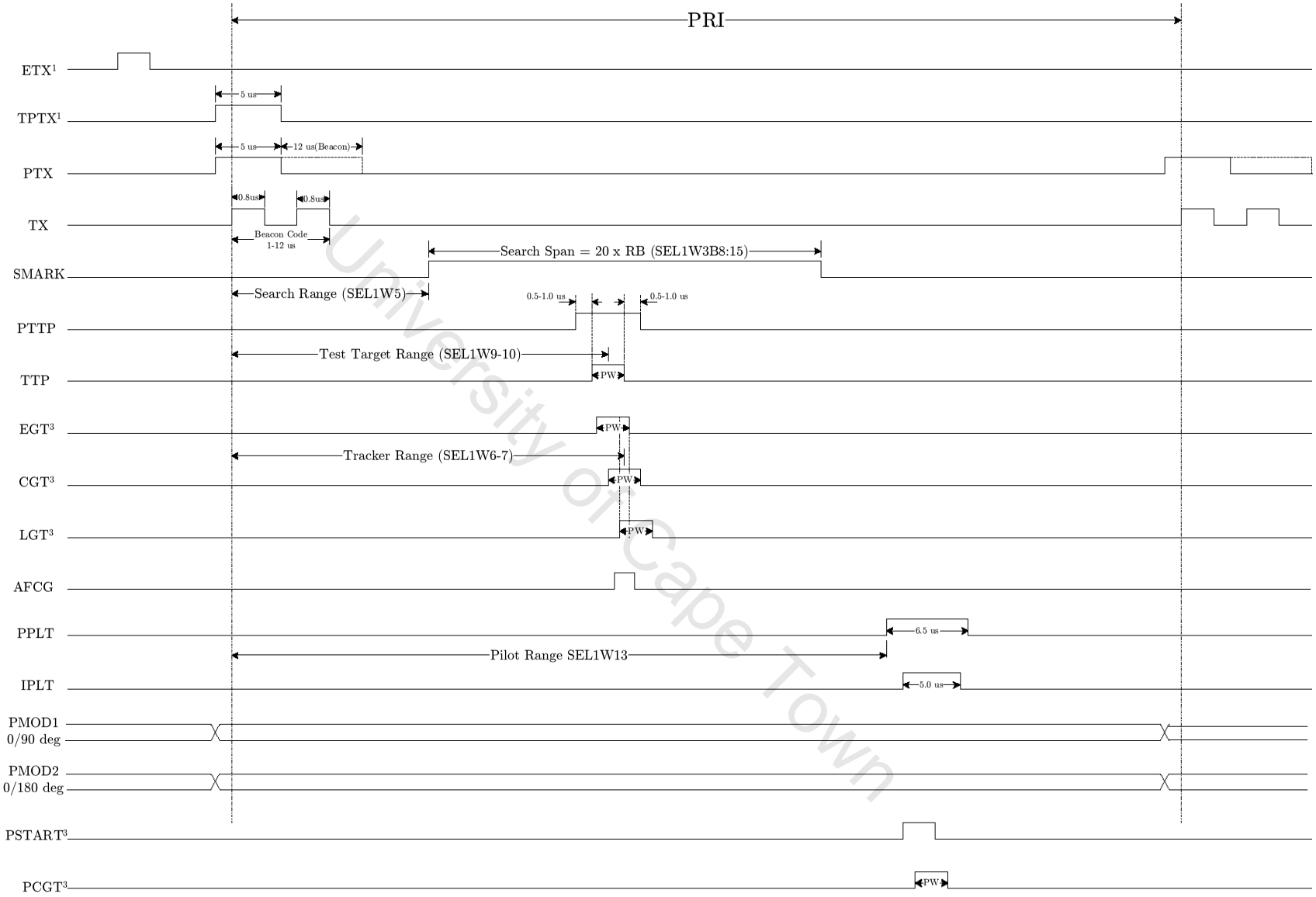
Pilot region Constant signals used for pilot processing.

It must be noted that, ideally, for target processing 32 *active* pulses are required [9]. The only exception to this is for long ranges when the PRF is low. In this case the duration of the transmit frame exceeds its limit ($RF_l > 20$ ms) and the radar transmits less than 32 *active* pulses. The minimum number of *active* pulses required for target processing is 16.

3.2.2 Synchronization Pulses per PRI

The radar frame, presented in the previous section, is 'constructed' through several synchronization (sync) pulses generated by the SVCU [2]. These sync pulses can be grouped into two time scales, per PRI and per frame.

The syncs produced per PRI, shown in Figure 3.2, govern the following tasks [20] [9]



1. Only occurs once per transmit frame
3. Internal to the SVCU

Figure 3.2: SVCU syncs per PRI.

Pulse transmission

The pre-transmit(PTX) and transmit (TX) syncs control the transmission of RF pulses by the radar, and hence also set the PRI/PRF. The PRI is the interval between two TX syncs in a radar frame.

The pulse width of PTX varies between 5 - 17 μs, depending on the mode the TR is operating in. The different operating modes of the TR are discussed in Section 3.3 below. The TX sync pulse is the width of the transmit pulse τ.

PTX is output to the RFH (generate TX drive) and to all the radar sub-assemblies that require a start of PRI sync. The TX pulse is only output to the transmitter to synchronise the output of the RF transmit pulses. Both these syncs are present in all L PRIs of a transmit frame.

Target video processing

The timing, per PRI, of the processing (sampling and data manipulation) of the target videos (ΣD, ΣT, ΔAZ and ΔEL) is controlled by the search marker (SMARK) and the three tracking gates (EGT, CGT and LGT).

The position of the SMARK is predetermined by the SEL and sent to the SVCU at the beginning of the frame. This sync invokes a state machine within the ADC Controller Card (see Section 4.3.1) that digitizes the ΣD video channel into 20 range bins (RB). A range bin within the TR is defined as $\tau \times 0.8$, and 20 of these form the search span (detection window) of the TR.

The EGT, CGT and LGT are placed around the expected time of arrival (calculated to within 6.25 ns by SEL) of the target return. These syncs, matched to the transmitted pulse width τ, integrate and hold (I/H) the ΣT, ΔAZ and ΔEL videos in the Measurement Card. These voltages are then digitized as described in Section 4.3.2.

The target video processing only occurs in the N *active* PRIs in the receive frame.

Frequency control

When the radar is operating in beacon mode, the frequency of the target echo is set by a beacon/transponder. In this mode the TR uses a VCO to mix down the received pulse to an intermediate frequency (IF).

In beacon mode, the AFC sync is output to the RFH to sample the ΣT video at the expected target position. Using this measurement the SEL ensures that the VCO frequency tracks the transponder frequency. This is necessary since both the VCO and transponder frequencies may drift during operation.

The AFC sync is output for every *active* PRI in the receive frame.

Pilot injection and processing

At the end[3] of every *active* PRI (N) a pilot signal is injected into the RF Front End (RFFE). The generation and injection of this pilot signal is controlled by syncs inject pilot (IPLT), pilot centre gate (PCGT) and pilot mod 1 and 2 (PMOD1/2).

The IPLT sync is sent to the RFH to generate the pilot pulse as described in Section 2.3.2. The PMOD1/2 syncs are used to introduce a phase shift (pseudo Doppler) into the pilot signal.

The pilot is sampled using the PCGT sync which invokes the ADC Controller card within the SVCU. The pilot is sampled in the middle of the IPLT sync because the delay down the RF chain is minimal.

The pilot signal is only used in the N *active* PRIs of the radar frame.

3.2.3 Synchronization Pulses per Frame

The TR transmit frame comprises L PRIs as described in Section 3.2.1. Each PRI contains the syncs characterised in the previous section. In addition to this, there are events in the TR that occur per frame. The syncs produced to control these events are presented below.

The syncs produced per frame, shown in Figure 3.3, govern the following events [20] [9]

Pre-frame syncs

The pre-frame syncs occur once per frame at the start of the frame (preceding first PRI of filling frame). They are the time zero trigger (t_0) and early transmit (ETX) syncs.

The t_0 trigger starts the SVCU, and hence the radar frame, at a known time. If the TR

[3] If the search span (SMARK) is close to the end of the PRI, the pilot is injected prior to it.

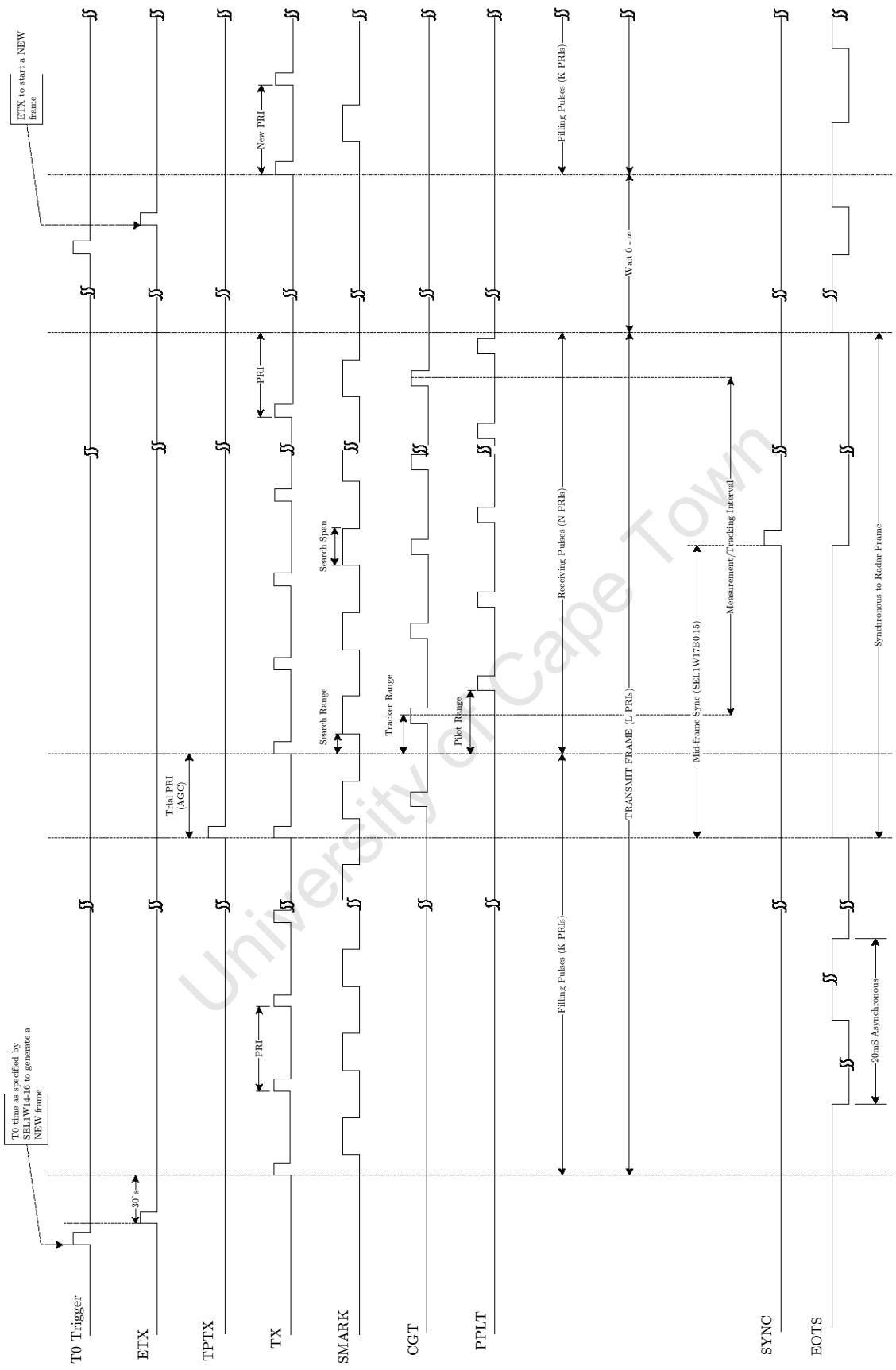


Figure 3.3: SVCU syncs per radar frame.

is producing consecutive transmit frames, the t_0 trigger only occurs in the first frame. This is discussed in Section 3.3.1.

The ETX sync occurs simultaneously with the t_0 trigger. When the TR is producing consecutive transmit frames (no t_0 trigger), ETX occurs 30 μs before the start of the next frame.

The ETX sync is sent to all the radar subsystems that require a start of frame reference.

Gain control

The TR has a high dynamic range [1]. This is achieved by controlling the gains of the receive channels, amplifying weak signals and attenuating strong ones.

The TRLAs (RFFE) form the first line of protection, for the receive channels, from strong signals. The attenuation value of the TRLA (0 - 63 dB in steps of 1 dB) is set by the automatic gain control (AGC) procedure run by the SEL. Consequently the AGC has a time constant of two cycles (radar frames) and cannot react immediately to very large signals at the receiver input [1].

To overcome this problem the TR uses an 'idle' pulse to invoke a saturation control (STC) function [7]. If the STC detects a strong signal (above a pre-defined threshold), a 20 dB attenuation is added automatically and is effective for the next *active* pulse of the same frame.

The 'idle' pulse sync, temporary pre-trigger(TPTX), is sent to the RFH at the beginning of the last PRI in the filling frame .

Optical syncs

The electro-optical tracker synchronisation (EOTS) sync is output by the SVCU to the optical system (TV Tracker) [20]. This sync is used to synchronise the optical frames to the radar frame.

In normal mode the EOTS is a free running 50 Hz square wave. If the radar is in Star Calibration mode, the EOTS is synchronized to the radar PRI.

Mid-frame sync

The mid-frame sync (SYNC) is generated at the middle of the measurement interval [20]. The measurement interval is defined as the time between the first and the last centre measurement gate in a frame (see Figures 3.1 and 3.3).

The SYNC is used in the TR for two purposes: (1) it is used by the SVCU, internally, to capture the mid-frame time from the STCG (see Section 4.5.1); and (2) it is sent to the pedestal subsystem to capture the pedestal sensor (azimuth and elevation) readings at this point.

3.3 TR Modes of Operation

When in 'mission mode' (tracking an aircraft), the TR can be operating in one of four modes. This is defined by two degrees of freedom: time and frequency.

In the time domain the radar can either be in independent or chain mode [1]. In the frequency domain the radar can either be in beacon or skin mode.

3.3.1 Independent and Chain Mode

There are three tracking radars at Denel Overberg Test Range. These radars can be required to operate simultaneously, at the same frequency (channel), all possibly tracking the same target. In such a scenario, the TRs must be synchronized with each other to prevent mutual interference.

This inter-radar synchronization is achieved by synchronizing each TR to a central atomic clock, via an RF IRIG B time code (STCG see Section 4.5.1). This allows the TR to run in one of two modes:

Independent mode

In independent mode the TR signal processing (SP) assumes that it is operating independently (only radar transmitting). The TR thus produces new radar frames as required, with no restrictions on the t_0 times of the frames being output. In independent mode the TR can produce two types of frames (set by SEL1W18B3) [20]:

Continuation frames The radar transmits (SVCU generates) the first frame at time t_0, and then produces consecutive frames with identical parameters (PRF, τ, search span etc.) until a new frame message is received from the SEL.

In continuation frames, the t_0 trigger is only present in the first frame, subsequent start of frames are denoted by the ETX sync.

New frames In this mode every frame is started by the t_0 trigger, at the time specified in the SEL message to the SVCU (SEL1W14:16).

Chain mode

In chain mode the SEL uses time division multiplexing to ensure that interference between TRs is avoided. Namely, each radar transmits its frame and then it is 'silenced' for the time required for the other radar(s) to transmit and receive the echo from the target. This is implemented by the SP in its calculation of the t_0 time for the TR.

All frames produced in chain mode are new frames.

3.3.2 Skin and Beacon Mode

The TR detects and tracks targets based on the echo received to a pulse previously transmitted at the target. This echo signal can either be a reflection of the transmit pulse, or a pulse transmitted by a beacon/transponder on the target [7].

Beacon mode

In beacon mode the TR is configured to detect/track a target with a transponder on it.

The transponder receives the TR pulse at frequency f_c, performs a frequency shift and retransmits the pulse at frequency f_{trans} and power $P_{t_{trans}}$. During the set up of the radar, the frequency shift amount, $\Delta f = f_c - f_{trans}$, is input by the operator. The inherent delay of the transponder is calculated, by the SEL, during a beacon calibration routine [1]. These two parameters are used by the TR signal processing.

As shown in Section 2.3.2 the transmit pulse frequency is set by the C-band synthesizer in the RFH (SYNTH-C). Due to the frequency shift, Δf, introduced by the transponder, in beacon mode the down-convertion of the received signal to IF1 is done by a VCO. The frequency of this VCO is set at

$$f_{\text{VCO}} = f_{\text{SYNTH-C}} - \Delta f \tag{3.10}$$

This ensures that the frequency at IF1 is maintained at 1080 MHz, regardless of whether the TR is operating in beacon or skin mode.

The VCO frequency is not phase locked to the MOSC and introduces an unknown phase component into the down-converted echo signal. This results in the inability to resolve phase changes in the echo that are due to the target motion. For this reason, beacon mode in the TR is *non-coherent*.

Double pulse Some transponders require two pulses in quick succession to function correctly. The first pulse 'wakes the transponder up' and it responds to the second pulse as described above. To cater for this, the SVCU can output two transmit pulses (sync TX) separated by a variable delay.

Skin mode

In skin mode all echoes arriving at the radar are assumed to be reflections of the transmit pulse. This enables the use of the SYNTH-C for up-conversion of the transmit pulse and down-conversion of the received echo.

Given the SYNTH-C is phase locked to the MOSC, the phase changes in the received signal are only due to the target motion. Skin mode in the TR is thus essentially *coherent*. Nonetheless, the SP may choose to ignore the phase information and operate in *non-coherent* skin mode [10].

3.4 SEL Signal Processing Cycle

The SEL signal processing cycle (SP-cycle) consists of a set of computation algorithms that serve as system control and signal/data processing tasks. The SP-cycle is synchronized to the radar frame by syncs ETX, RTC 20 ms and RTC 1 s.

The timing is loosely given by: SP-cycle (n) processes the data from the L PRIs in previous transmit frame $(n-1)$ and uses this to compute the parameters for the next frame $(n+1)$.[†]

[†]The TR can run in a two or three frame (cycle) mode depending on which mode (independent/chain)

The previous, current and next frames/cycles are shown in Figure 3.4 [19] [9].

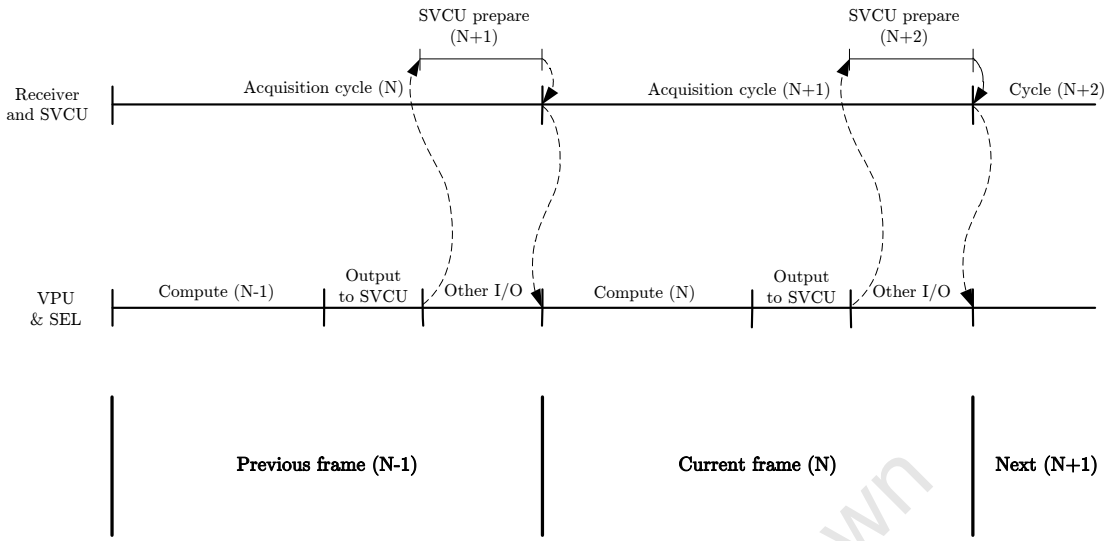


Figure 3.4: SVCU and SP cycle sequence.

3.5 Target Detection and Acquisition

The TR uses the detection channel, ΣD(I,Q) videos, from the receiver to detect potential targets [7]. The main characteristics of the sampling, and subsequent processing, of the detection channel is described here. For a complete description of the TR signal processing please see [10].

3.5.1 Sampling Strategy

The analogue detection videos output by the receiver are sampled (digitized) within the Detection Card of the SVCU [2]. The I and Q videos are sampled in an identical manner.

The detection channel is sampled at a constant 10 MHz using a 9 bit ADC. The sampling occurs for a fixed duration (knows as the detection window or search span), around the expected time of arrival of the target return, which is called the "tracker" range [1].

The digital samples are summed to form M=20 overlapping range bins (gates) [10]. The range bins are spaced 0.8 × PW apart, with an overlap factor of 0.2 × PW, where PW

the radar is operating in. See [22].

is the TR's pulse width (0.5, 1.0, 2.5 or 5.0 μs). The resulting range bin values are also corrected for gain and phase errors as described in Section 4.3.4.

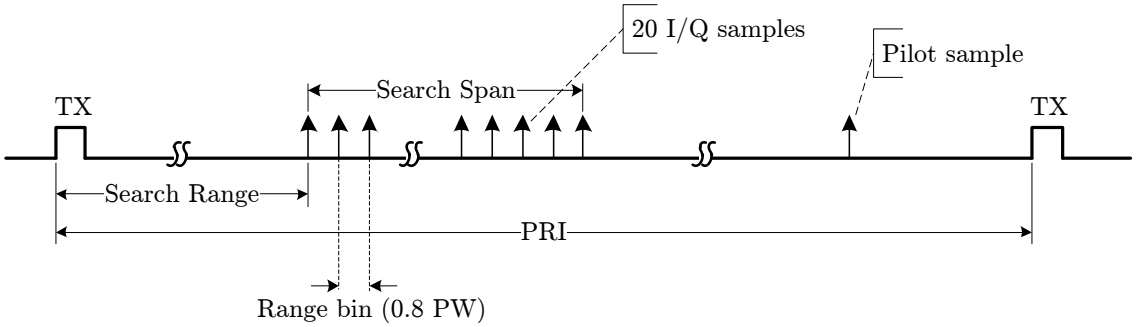


Figure 3.5: The detection channel sampling strategy.

For every radar frame, with N *active* PRIs, the SVCU passes an array $x(n, m)$ of $N \times M$ complex (I/Q pair) numbers to the system computer for detection.

3.5.2 Pulse Doppler Processing

The sampling process (within SVCU) does not distinguish between the *coherent* and *non-coherent* situations [10]. In the signal processing a distinction is made.

Coherent integration

If the received signal is *coherent* (TR RFH set up in *coherent* mode see Section 2.3.2), it is possible to perform *coherent* integration on the detection samples [10]. The purpose of this integration is to increase the signal to noise ratio (SNR) and to separate faster moving targets from the clutter [5].

Coherent processing is done by performing a 32[†] point DFT on the N samples of each range gate. This is implemented in the TR signal processing by [10]

$$Y_D(k, m) = \sum_{n=0}^{L-1} w(n; N)x(n, m)e^{-j2\pi nk/L}, \qquad \begin{matrix} k = 0, ..., L-1 \\ m = 0, ..., M-1 \end{matrix} \tag{3.11}$$

[†]This is independent of the number of *active* pulses N. If less than 32 *active* pulses are transmitted, the data is padded with zeroes.

where $L = 32$ is the number of DFT points and $M = 20$ the number of range bins. $w(n; N)$ is a complex weighting function (window) for sidelobe suppression described by [10] [5]

$$w(n; N) = w_r(n; N)e^{-j2\pi n(n-N-1)F_a}, \quad n = 0, ..., L-1 \tag{3.12}$$

where F_a is a factor resulting from the expected radial acceleration of the target and $w_r(n; N)$ is the real weighting values given by

$$\begin{aligned} w_r(n; N) = &\quad w_r(N-1-n; N), \qquad n = 0, ..., N-1 \\ = &\quad 0, \qquad\qquad\quad n = N, ..., L-1 \end{aligned} \tag{3.13}$$

normalized so as to obey

$$\sum_{n=0}^{N-1} w_r(n; N) = 1 \tag{3.14}$$

The absolute value of $Y_D(k, m)$ is taken which results in [10]

$$Z_D(k, m) = |Y_D(k, m)|^2 \tag{3.15}$$

a 20 by 32 matrix, where each element contains the power in each range-Doppler cell.

Non-coherent integration

In the *non-coherent* case the phase is disregarded. The absolute value of the samples belonging to each range gate are summed [10]

$$Z_D(m) = \frac{1}{N}\sum_{n=0}^{N-1} |x(n, m)|^2, \qquad m = 0, ..., M-1 \tag{3.16}$$

where $M = 20$ is the number of range bins and $16 \leq N \leq 32$ the number of *active* pulses in the frame. The result is a 20 element vector containing the power of the signal in each of the range gates.

3.5.3 CFAR Detector

In *coherent* mode a preliminary detection (pre-det) is established by comparing each cell in $Z_D(k, m)$ to a threshold $D(k)$.

Thresholding

A threshold value, per Doppler filter (one row of range gates in the $Z_D(k, m)$ range-Doppler power map) , is adaptively calculated by averaging the 15 weakest range cells in that Doppler filter. For the k^{th} Doppler filter the threshold [10]

$$D(k) = K^2(k)[p_1 + \frac{1}{15}\sum_{i=1}^{15} Z_D(k, i^*)], \qquad k = 0, ..., L-1 \tag{3.17}$$

where $Z_D(k, i^*)$ is a row in the range-Doppler map sorted in order of ascending power value; p_1 is a constant preventing $D(k)$ from being zero; and $K^2(k)$ is the threshold constant corresponding to the k^{th} Doppler filter. This threshold constant depends on the expected false-alarm probability (P_{FA}).[4]

The adaptive threshold is calculated for *coherent* detection to provide a constant false alarm rate (CFAR) [1] [10]. The TR detector is therefore known as a CFAR or cell averaging CFAR (CA-CFAR).

Clustering

The CFAR detector assigns a preliminary detection (pre-det) to all cells $Z_D(k, m) > D(k)$. Because the target is not a point target, it will usually appear as a group of 'neighbouring' pre-dets. Two pre-dets, (k_1, m_1) and (k_2, m_2), are considered neighbours if [10]

[4]Radar detection is a probabilistic exercise based on the likelihood of a target echo being present in the received signal. A common figure of merit of a radar detector is the probability of false alarm P_{FA} [4]. Most radars operate at a threshold level giving very low false alarm rates ($P_{FA} < 10^{-6}$).

$$
\begin{aligned}
|m_1 - m_2| = 1 \quad and \quad k_1 = k_2 \\
or \\
|k_1 - k_2| = 1 \quad and \quad m_1 = m_2
\end{aligned}
\tag{3.18}
$$

A set of pre-dets are termed a cluster G if each pre-det has at least one neighbour and the cluster does not exceed the maximum permitted dimensions in range and Doppler.

The centre of gravity, (k_c, m_c), of each cluster G is calculated by centroiding [5] [10]

$$
\begin{aligned}
k_c = \quad & \frac{\sum k Z_D(k,m)}{\sum Z_D(k,m)} \\
m_c = \quad & \frac{\sum m Z_D(k,m)}{\sum Z_D(k,m)}
\end{aligned}
\tag{3.19}
$$

for all the cells (k, m) that belong to cluster G.

Course ambiguous range and Doppler

The course ambiguous range and Doppler for the detected target is finally given as [10]

$$
\begin{aligned}
R_{amb} &= R_{detwin} + (m_c)(0.8 \times \text{PW})(\frac{c}{2}) \\
F_{d_{app}} &= \frac{k_c}{L}(\text{PRF})
\end{aligned}
\tag{3.20}
$$

where R_{detwin} is the range to the start of the detection window (seach span) and L is 32, the number of points in the FFT.

3.5.4 Non-coherent Detector

In *non-coherent* detection a similar thresholding algorithm is implemented on $Z_D(m)$, the 20 element output of the *non-coherent* integration.

All values of $Z_D(m) > D$ are assigned as preliminary detections (pre-dets) and grouped into clusters exactly as in the *coherent* case. Here a cluster is a set of consecutive range gates declared to have pre-dets.

The centre of these clusters are calculated as

$$m_c = \frac{\sum m Z_D(m)}{\sum Z_D(m)} \tag{3.21}$$

for all the cells (m) that belong to a cluster. The ambiguous range is then calculated as per Equation 3.20.

3.5.5 Plot Extraction

The TR keeps track of the targets using a track while scan (TWS) procedure [10]. The TWS associates, if possible, targets detected in the current cycle to targets detected in previous cycles to form tracks. Tracks to which no new clusters have been associated for several cycles are also deleted.

If the radar is not tracking, the target closest to the centre of the range-Doppler map is passed for tracking [1]. The TR changes the PRF in order to verify the range of the target, and the magnitude of the angle difference channels (ΔAZ and ΔEL) are also checked to ensure the target is within the antenna beam.

At this point the TR tracking algorithm is initiated and the TR commences with *non-coherent* tracking [1].*

3.6 Target Measurement and Tracking

The TR uses three tracking channels (ΣT, ΔAZ and ΔEL) to measure the position of a target in four axes: range, azimuth, elevation and Doppler [7]. The main characteristics of the sampling, and subsequent processing, of the tracking channels are described here. For a complete description of the TR signal processing please see [10].

*Tracking within the TR always begins *non-coherently* even in *coherent* radar operation [10].

3.6.1 Sampling Strategy

The three analogue tracking channel videos output by the receiver are sampled (digitized) within the Measurement Card of the SVCU [2]. All six video signals (three I/Q pairs) are sampled in an identical manner.

The ΣT, ΔAZ and ΔEL videos are sampled simultaneously at the expected time of arrival of the target echo ("tracker" range).[5] This is known as the centre sample or centre gate [10]. For range tracking, the ΣT channel is also sampled slightly earlier (early gate) and later (late gate) than the "tracker" range [1]. The resulting measurement samples are also corrected for gain and phase errors as described in Section 4.3.4.

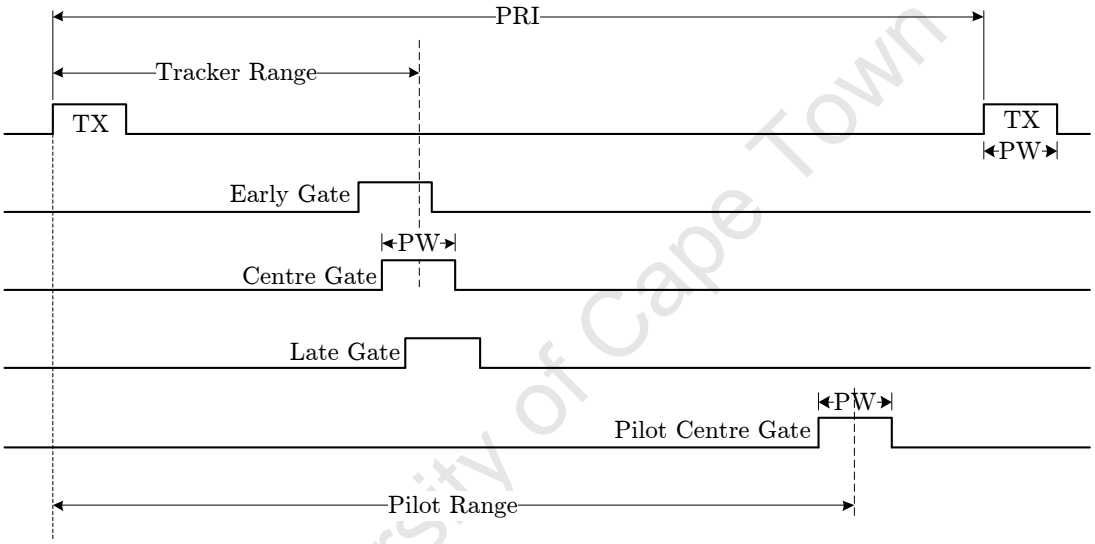


Figure 3.6: The tracking channel sampling strategy.

For every radar frame, with N *active* PRIs, the SVCU passes an array $x(n, l)$ of N groups of five complex (I/Q pairs) samples to the system computer for measurement and tracking.

3.6.2 Pulse Doppler Processing

As with the detection process, the samples obtained from the SVCU can be processed *coherently* or *non-coherently* [10].

[5] The sampling process is by means of a set of integrate and hold circuits as described in Section 4.3.2.

Coherent integration

In *coherent* mode the five complex samples are *coherently* summed by means of a N point DFT, implemented as [10] [5]

$$Y_T(l) = \sum_{n=0}^{N-1} w(n;N)x(n,l)e^{-j2\pi nF'_n}, \qquad l = 1,2,3,4,5 \tag{3.22}$$

where F'_n is a sum of the expected Doppler frequency F_D and a factor due to the radial acceleration of the target given as

$$F'_n = F_D + (n - 2(\frac{N+1}{2}))F_a \tag{3.23}$$

In addition to this the $\Sigma\mathrm{T}_C$ ($l = 1$) is also integrated for a frequency slightly below and above the expected Doppler, as

$$\begin{aligned} Y_T(6) &= \textstyle\sum_{n=0}^{N-1} w(n;N)x(n,1)e^{-j2\pi n(F'_n+\Delta F_D)} \\ Y_T(7) &= \textstyle\sum_{n=0}^{N-1} w(n;N)x(n,1)e^{-j2\pi n(F'_n-\Delta F_D)} \end{aligned} \tag{3.24}$$

$Y_T(4)$ and $Y_T(5)$ ($\Sigma\mathrm{T}_E$ and $\Sigma\mathrm{T}_L$) are used for target range estimation (error of range prediction); $Y_T(6)$ and $Y_T(7)$ are used to calculated the target Doppler frequency deviation from the predicted F_D [10].

Non-coherent integration

The *non-coherent* integration in the tracking channel is, as in the detection case, a summation of the amplitudes of the five sample sets. This is implemented as [10]

$$Y_T(l) = \frac{1}{N}\sum_{n=0}^{N-1} |x(n,l)|^2, \qquad l = 1,2,3,4,5 \tag{3.25}$$

For the *non-coherent* case $Y_T(6)$ and $Y_T(7)$ are not calculated as the target Doppler is not used.

3.6.3 Measurements

The measurement process, rather than measure the absolute position of the target, provides an estimate of the deviation (error) of the measured target position from its expected position in the four axes measured, i.e. range, Doppler, azimuth and elevation [10].

The error signal for each axes is formed by a monopulse technique [5]. Namely, in each dimension/axes, the real part of a sum to difference ratio is calculated [1]

$$\text{error}(x) = \text{real}\left(\frac{\Delta(x)}{\Sigma(x)}\right) \tag{3.26}$$

Range

The error signal in range is approximated by [10]

$$\varepsilon_R = K_R^{T(T)}\text{real}\left(\frac{Y_T(4) - Y_T(5)}{Y_T(4) + Y_T(5)}\right) \tag{3.27}$$

$Y_T(4)$ and $Y_T(5)$ are the integrated samples obtained from the previous processing stage. $K_R^{T(T)}$ is a factor, calculated during Range Calibration, to convert the electrical error to a physical error.

Doppler

The Doppler frequency is only measured in *coherent* processing. The error frequency is evaluated as [10]

$$\varepsilon_D = K_D^{B}\text{real}\left(\frac{Y_T(6) - Y_T(7)}{Y_T(6) + Y_T(7)}\right) \tag{3.28}$$

$Y_T(6)$ and $Y_T(7)$ are the integrated samples for a frequency on either side of the expected

Doppler of the target. K_D^B is a factor to convert the electrical error to an actual frequency error amount.

Azimuth and Elevation

The antenna feed horn provides the difference in azimuth ΔAZ and the difference in elevation ΔEL directly, with the sum ΣT common to both [7].

The SP uses the corresponding azimuth and elevation tracking samples ($l = 2, 3$) to calculate a rough angle error [10]

$$\theta_l = \text{real}\left(\frac{Y_T(l)}{Y_T(1)}\frac{1}{B(l, f_c, p)}\right), \qquad l = 2, 3 \tag{3.29}$$

where $B(l, f_c, p)$ is a factor, dependent on the carrier frequency and antenna polarization, to convert the electrical error to an angular value.

3.6.4 Tracking and Prediction

The main objective of the TR is to perform accurate and reliable tracking of a target [1] [10]. The TR uses the Kalman filter algorithm to smooth (correct) and predict the target state vector (position in range, velocity, azimuth and elevation).

The mathematics of the Kalman filter are well documented [23] [5], and only a top level description of how it is applied in the TR is presented here. For details on the TR SP pertaining to tracking see [10].

The TR tracking algorithm entails smoothing and prediction. A new set of measurements from the SVCU are processed as described in Sections 3.6.1 - 3.6.3 and input into the tracking filter. The filter then corrects (smooths) the previously predicted state vector based on these new measurements [10].

This smoothed target state is then used by the tracking filter to predict the state of the target in the next processing cycle (new frame). The SEL prepares the various radar sub-systems for the predicted state, by sending out appropriate MUXBUS messages. These include setting the antenna direction, t_0 time, setting the new tracker range and selecting the PRF for the new frame.

The new frame is then processed, and the smoothing, prediction process is repeated. This tracking cycle within the TR is shown in Figure 3.7.

3.7 Summary

The four time scales that exist in a pulsed radar system were presented and their role in the TR timing strategy discussed. The concept of a radar frame was then introduced. The various syncs, that occur per frame and per PRI, that constitute the radar frame were shown.

The detection process within the TR digitizes the ΣD(I,Q) videos and forms 20 overlapping range gates. The gate values are corrected for DC offsets and gain and phase errors. In *non-coherent* mode the absolute value of the samples are summed up for each gate, resulting in a 20 element power vector. In *coherent* mode a 32 point FFT is performed for each range gate and its absolute value taken. This results in a 32 by 20 range-Doppler matrix. The integrated vector/matrix is then passed through a CFAR detector and clustering algorithm to identify probably targets.

The measurement process estimates the error of the measured target position from its expected position in the four axes measured(range, Doppler, azimuth and elevation). To do this the ΣT, ΔAZ and ΔEL I/Q videos are sampled by means of three tracking/measurement gates: early, centre and late gates. In *non-coherent* mode the absolute value of the samples are summed up for each tracking gate sample pair. In *coherent* mode the values are integrated by means of a DFT. The range error is then determined by computing the delta-over-sum ratio of the early to late gate integrated samples. The Doppler (*coherent* mode only) is found by evaluating the centre gate sample at a frequency slightly above and below the expected Doppler frequency by means of a DFT. The azimuth and elevation errors are computed directly from the ΔAZ and ΔEL samples.

The measured parameters are then passed through a Kalman filter algorithm that smooths (correct) and predicts the target state vector (position in range, velocity, azimuth and elevation).

The next chapter investigates the architecture of the SVCU in closer detail. This will examine how the SVCU implements, on a hardware level, the various timing and sampling requirements of the TR discussed in this chapter.

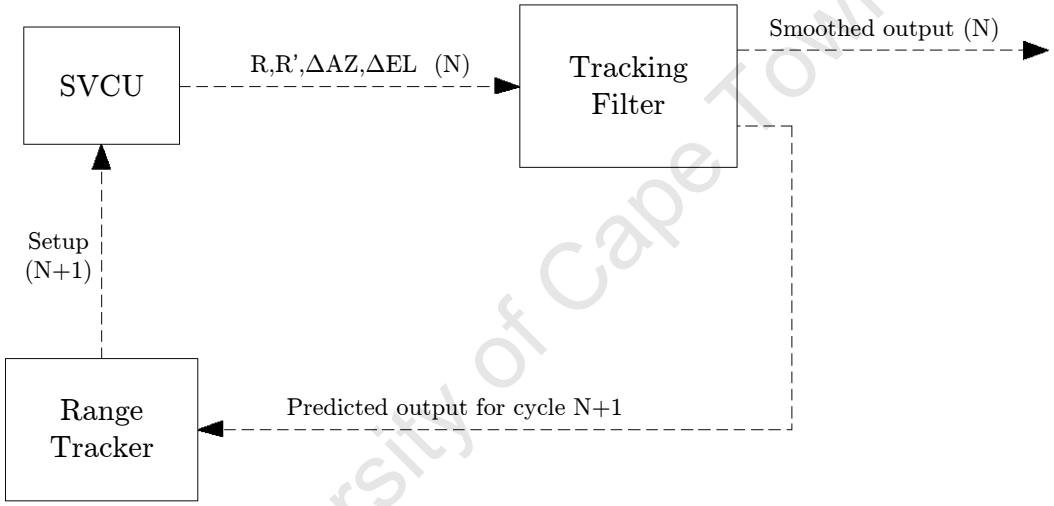


Figure 3.7: TR smoothing and prediction

Chapter 4

Synchronization and Video Conversion Unit

The SVCU is the focus of this dissertation, thus a comprehensive discussion of its hardware, and operation thereof, is presented in this chapter.

The function of the SVCU within the TR is first discussed, followed by a general description of the cards and modules that constitute the SVCU hardware. The four primary tasks of the SVCU: (1) digitizing the receiver video signals; (2) producing sync pulses for the radar sub-systems; (3) providing time within the radar; and (4) producing composite video for the A/R scopes are then discussed in detail.

4.1 Functional Description

Additional to the ten baseband video signals (from Receiver), the SVCU also receives the time of day from a Standard Time Code Generator (STCG) and a reference 10 MHz clock from the RF Head to synchronize the SVCU to the radar [2]. Communication with the main system computer is done through a High Speed Data Interface (HSDI) and the 1553 MUXBUS.

Using these inputs and outputs the SVCU performs the following functions:

- Sample and digitize the analogue, baseband video, signals from the receiver.
- Correct the digital data, convert it into a 32 bit floating point format and transfer it to the system computer (SEL).

- Synchronizes all the sub-systems of the radar, i.e. provide them with synchronization pulses and clocks.
- Start the SVCU at a predefined time (t_0). Capture the mid-frame time.
- Supply analogue video, with markers, to the A-scopes (CCA).

4.2 Hardware Description

The SVCU contains analogue and digital circuits including a microprocessor board (based on the iSBC 86/12). The SVCU is modular in design and is comprised of several cards and modules[1].

The various sub-units of the SVCU, including a exp(23) short description of the primary functions thereof, are provided below. See Figure 4.1 for a block diagram of the SVCU.

Video Receivers Card

Receiver channels ΣD(I,Q), ΣT(I,Q), ΔAZ(I,Q), ΔEL(I,Q) and ΣV1,2 are input into the SVCU via the video receivers. This card contains analogue circuitry to condition the signals and provide isolation between the SVCU and the radar receiver. It also contains a multiplexer to switch between the receiver signals and test signals.

Detection ADC Card

The detection A/D converts the ΣD(I,Q) signals into 9 bit digital data for detection functions. The detection A/D is operated by the A/D controller and converts on a 10 MHz clock (resolution of 100 ns). This data is then forwarded to the arithmetic unit.

Measurement ADC Card

The measurement A/D converts the ΣT(I,Q), ΔAZ(I,Q), ΔEL(I,Q) signals into 12 bit digital data for tracking functions. Although it is also operated by the A/D controller, the measurement A/D is able to position or 'track' to a resolution of 6.25 ns because

[1]The 'cards', in this context, were custom designed for the SVCU whilst the 'modules' were bought, and incorporated into the SVCU, by the manufacturers.

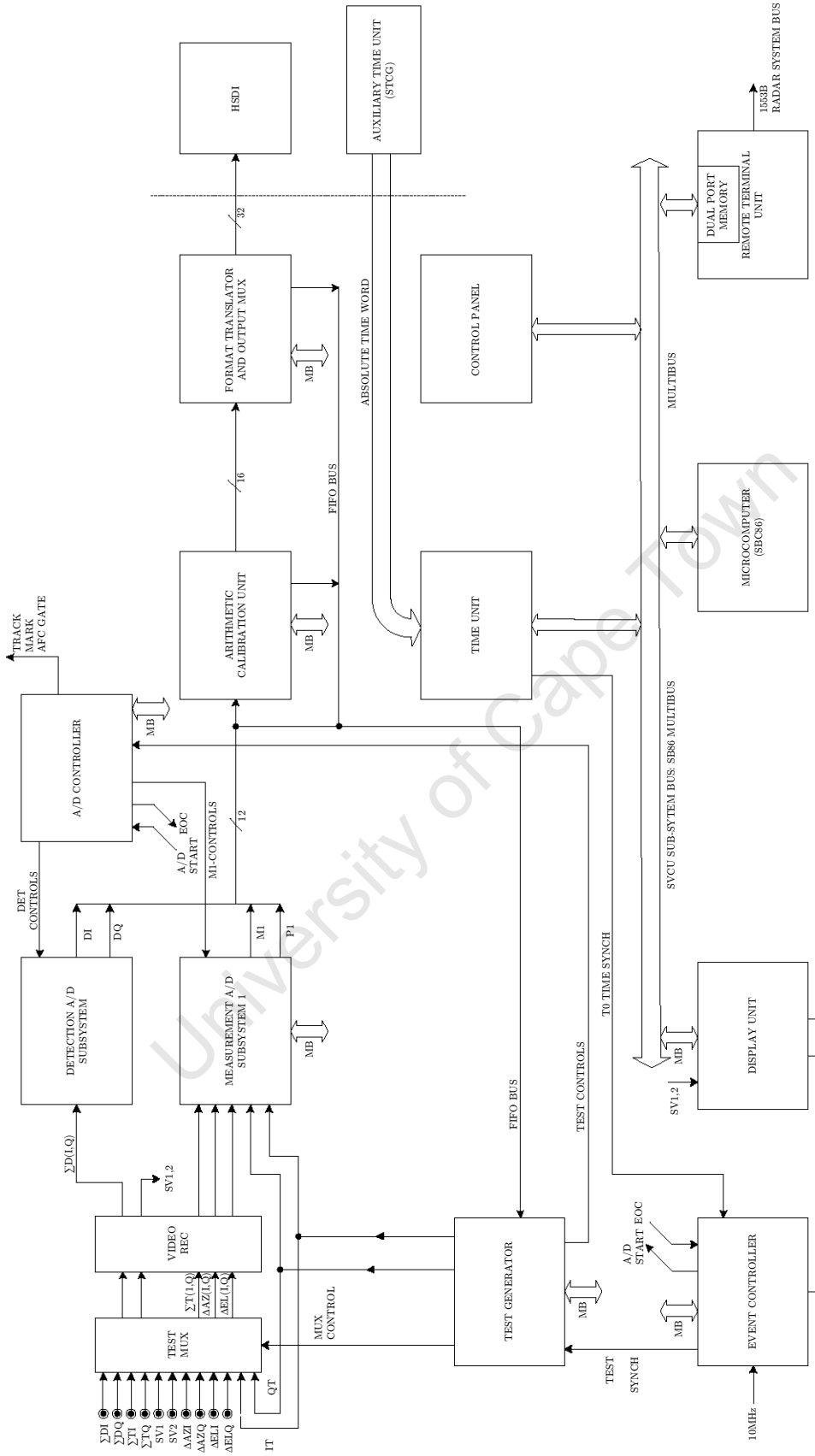


Figure 4.1: A block diagram of the SVCU.

of ECL module A which contains a 160 MHz clock [20]. The converted data is then forwarded to the arithmetic unit.

A/D Controller Card

The A/D controller contains all the logic to operate the measurement and detection A/D. The A/D controller is in turn operated by the event controller. The A/D controller also contains some logic to synchronize the digital data that is being sent to the arithmetic unit.

Arithmetic Calibration Card

The arithmetic unit performs data correction on the digital data in the form of: scaling, gain correction and offsets. The corrected samples have a precision of 16 bits and are sent to the format unit.

Format and Output Card

The format unit transforms the 16 bit integers to 32 bit floating point format. It then sends this data to the main computer via the HSDI.

Test Interface Card

The test generator performs built in tests to analyse and calibrate the analogue channels before system operation. It can perform tests whilst offline (test mode) or during one PRI per frame for real time calibrations.

Event Controller Card

The event controller produces the trigger pulses which activate the data conversion processes as well as the transmitter and RF sections. The event controller starts when it receives the t_0 trigger from the time unit [9].

Time Unit Card

The time unit compares the absolute time from the STCG with a time previously loaded by the microprocessor (SBC86) and provides the t_0 trigger to the event controller. It also allows the iSBC86 to read the absolute time to within an accuracy of 1 μs.

Single Board Computer Card

The Intel Single Board Computer 86/12 (iSBC86) receives its commands and parameters from the system computer via the RTU. The primary function of the iSBC86 is to translate these system messages into hardware controls and load the applicable SVCU units with data. The iSBC86 is also involved in calibrating the ADCs and reporting faults to the CCA.

Panel Driver Card

The panel driver provides hardware interface from the SVCU front panel to the display unit.

Display Unit Card

The display unit interfaces the radar with the A/R-scopes display by providing skin and beacon composite video. It also controls the front panel display functions and contains watchdog circuits.

ECL Module A

ECL A contains a phase-locked loop used to synchronize the 10 MHz reference clock from the RF Head. It also has a fine time delay and delayed clock to position the measurement A/D to within 6.25 ns.

ECL Module B

ECL B only contains a fine time delay and delayed clock. This module is used to position a test target pulse when required (calibration). Both ECL modules are controlled by the event controller.

Remote Terminal Unit (RTU)

The RTU interfaces the iSBC86 and hence the SVCU with the 1553 MUX Bus , also known as the system bus. Messages from the system computer are stored in the RTU memory and are read by the iSBC86 each frame.

4.3 Digitization of Video Signals

Digitizing the video signals from the receiver is not as straightforward as clocking an ADC. The A/D conversions have to be done at specific times in accordance with the transmit frame and radar syncs.

The video signals associated with detection and those associated with tracking are converted via two separate channels/streams. This is because they have different pre and post conversion (A/D) requirements (A/D timing, signal conditioning, digital manipulation etc.). The converted data must also be corrected and formatted before it can be sent to the master radar computer (SEL).

4.3.1 Detection Channel

The detection channel is responsible for the digitization of the ΣD(I,Q) video signals. This D/A conversion process is currently implemented the SVCU Detection Card. The Detection Card consists of two identical channels, for the I and Q video signals.

Twenty (20) I/Q sample pairs are converted per PRI as shown in Figure 4.2. These digitized samples are used by the SEL for target acquisition and plot extraction (see Section 3.5 for details on the associated signal processing). The main components of one channel is discussed below.

Input amplifier

Each video channel (I and Q) contains an input conditioning stage. This converts the input signals from ±2.5 Vp-p to the range of the ADC, -2 V to 0 V. This is done with the use of an amplifier with gain = -0.4 and then adding an offset voltage of -1 V.

A bandwidth limiting capacitor is also used to set a high frequency roll-off at 3.5 MHz. This prevents high frequency noise and harmonics from causing A/D conversion errors.

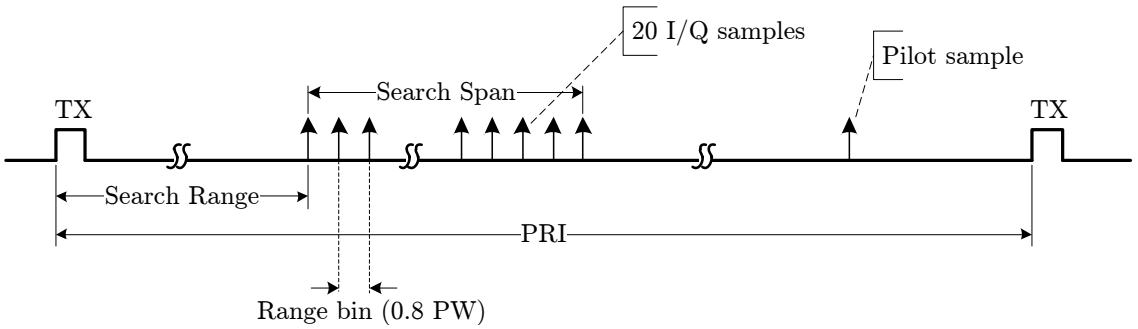


Figure 4.2: The detection channel sampling strategy.

This also allows for the use of a flash converter without a sample and hold amplifier preceding it [20].

A/D Converter

The A/D conversion is then done by two 9-bit monolithic 'flash' converters, one each on channels I and Q. The conversion takes place on the rising edge of DETCONV, a synchronized 10 MHz clock signal. These samples are then filtered by the next section.

Digital integrator and attenuator

Although the ΣD(I,Q) video signals are being sampled continuously at 10 MHz, only a small portion of it is utilized by the SEL, per PRI, for detection purposes. This is the search span of the radar (known as the detection window) and is illustrated in Figure 4.2.

The search span is further split into twenty (20) range bins. Each range bin is of width RB = 0.8 × PW. The RB value is calculated by summing the samples equivalent to one PW using a digital integrator. Summing the video for the duration of the PW effectively correlates the received signal to the transmit signal since the transmit signal is a simple pulse (rectangular). This correlation process (filter) increases the SNR [1] [4]. The signal processing pertaining to detection is dealt in greater detail in Chapter 3.

The number of samples integrated to form the RB varies from 5 (PW = 0.5 μs) to 50 (PW = 5 μs). This is because the ΣD video is sampled at a constant 10 MHz. To compensate for the RB value varying as the PW changes, a shift attenuator (left or right shift data

word) is placed between the ADC and the digital integrator. The effect of this attenuator and its output is demonstrated in Table 4.1 and 4.2.

Table 4.1: Output of the 12 bit attenuator per PW.

PW (μs)	**12 bit atten out (offset binary)**												**Max**
0.5	S	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	$7F8_{16}$
1.0	S	S	D7	D6	D5	D4	D3	D2	D1	D0	0	0	$3FC_{16}$
2.5	S	S	S	D7	D6	D5	D4	D3	D2	D1	D0	0	$1FE_{16}$
5.0	S	S	S	S	D7	D6	D5	D4	D3	D2	D1	D0	$0FF_{16}$

Table 4.2: Final RB value per PW.

PW (μs)	**No of samples**	**Max 12 bit atten out**	**RB value**
0.5	5	2040 ($7F8_{16}$)	10200
1.0	10	1020 ($3FC_{16}$)	10200
2.5	25	510 ($1FE_{16}$)	12750
5.0	50	255 ($0FF_{16}$)	12750

The output of the attenuator combined with the corresponding samples per PW produces a relatively constant range of numbers.

These values are then sent, via a FIFO bus, to the Arithmetic and Format Unit for data correction and formatting.

4.3.2 Tracking Channel

The tracking channel is responsible for the digitization of the ΣT(I,Q), ΔAZ(I,Q) and ΔEL(I,Q) video signals. The conversion process is done using three gates as shown in Figure 4.3.

Ten (10) I/Q sample pairs are converted per PRI. These samples are used for tracking, measurement and prediction of the next target position/vector. This section explains the D/A conversion process as implemented in the current SVCU Measurement Card. The primary components of this card are:

Input Attenuator

The input signals first pass through an attenuator to match the integration gain to the pulse width. A ratio that is an inverse of the PW is taken to ensure the integrator output

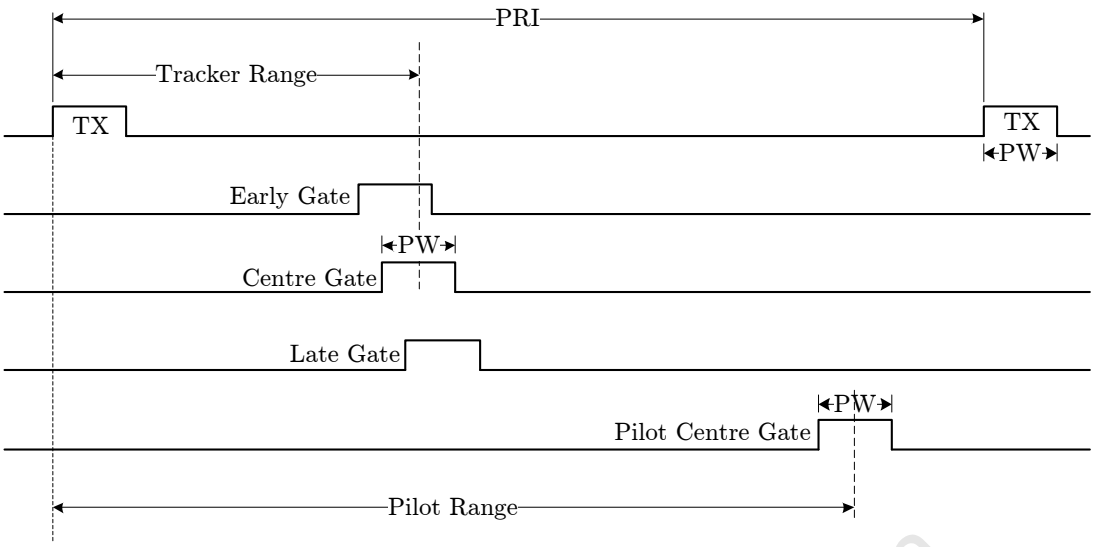


Figure 4.3: The tracking channel sampling strategy.

stays within the nominal ±2.5 V.

Integrate and Hold Circuits

The signal from the input attenuator is then amplified by a gain of 4 resulting in a ±10 V signal. This is done to increase the size of the components (resistors and capacitors) used to set the gain of the integrate-and-hold (I/H) circuits to a reasonable value. This gain is set as $-\frac{1}{RC}$.

There are three groups of I/H circuits: early, centre and late. These are controlled by the early gate (EGT), centre gate (CGT) and late gate (LGT) sync signals. All the gate widths are identical and matched to the PW (0.5, 1.0 and 2.5 μs).

Six centre I/H circuits integrate the ΣT(I,Q), ΔAZ(I,Q), ΔEL(I,Q) signals. The ΣT(I,Q) signals are sampled additionally on two early I/H and two late I/H circuits. The early and late sampling of the ΣT channel is to allow for range tracking by the SEL. This is discussed in Section 3.6.3.

The ten I/H samples are then passed to an analogue multiplexer. At the input to the multiplexer the signal gain is as follows:

$$\begin{aligned}\frac{V_{out}}{V_{in}} &= \text{input atten} \times \text{amplifier} \times \text{integrator} \\ &= (\text{PW} \times 10^{-6}) \times \frac{0.5}{\text{PW}} \times 4 \times -\frac{1}{\text{RC}} \\ &\quad \text{RC} = 2 \times 10^{-6} \\ &\quad \text{PW} = 0.5, 1.0, 2.5 \\ &= -1\end{aligned} \tag{4.1}$$

This keeps the output to the analogue multiplexer constant regardless of the changing PW. Note that the polarity of the signal has been changed due to the negative gain.

Analogue Multiplexer

An analogue multiplexer (MUX) is needed because there are ten signals to be digitized and only one ADC on the card. The MUX selects each of the ten I/H signals sequentially based on a four bit address received from the A/D Controller Card.

Calibration Circuits

The signal selected by the MUX first passes through some calibration circuitry, to perform gain and offset corrections, before it reaches the ADC. The corrections are done based on correction factors stored in the calibration memory. These factors[2] are calculated, by the iSBC86, when an SVCU calibration is performed.

The gain errors are corrected by means of an attenuator, and the offset errors using a summing op-amp. The attenuator gain and summing circuit value is determined by a DAC, which in turn is set by the values in the calibration memory.

Nominally the attenuator stage would have a gain of $\frac{1}{2}$, I/H circuitry gain of -1, and the offset circuit an offset of +2.5 V and gain of 2. This transforms the input signal from ±2.5 V to 0-5 V. This matches the dynamic range of the ADC.

[2]Correction factors for gain and phase errors within the radar receive channels as a whole are calculated during receiver channel calibrations, see [10].

A/D Converter

The ADC in the tracking channel is a 12 bit converter. Unlike the detection ADC, the tracking ADC converts on the rising edge of a positive going pulse (CONV). The CONV pulse can only be applied once the I/H, MUX and calibration circuits have been 'set up' correctly. The timing signals that controls this process is generated by the A/D Controller Card. Detailed timing diagrams of the Measurement Card are shown in [2] and a brief description thereof is given below.

The tracking gates (EGT, CGT and LGT) are applied to the I/H circuits. The multiplexer is switched to the first channel. A gain and offset calibration value, matching the current channel, is then loaded into the gain and offset DACs. Finally a positive going pulse onto the ADC convert pin converts the analogue voltage to a 12 bit digital number. The MUX is then switched to the next channel.

The calibration, conversion and incrementing channel process is then iterated. In one measurement cycle all ten analogue signals are digitized sequentially in this manner. These samples are then transferred, via a FIFO bus, to the Arithmetic and Format Unit for data correction and formatting.

4.3.3 Pilot Signal

The radar has the ability to inject a pilot signal into the RFFE. Because the pilot is of a known amplitude and phase, the pilot samples can be used for calibration and-or RT data correction purposes by the SEL [1].

The timing of the injection of the pilot is determined by the SEL (SEL1W13) and is implemented by the SVCU using sync pulse I-PLT (see Section 3.1 for more details).

Conversion

Pilot signals injected into the RFFE are sampled in the following manner (see Figures 4.2 and 4.3):

Detection channel One RB value (I/Q pair) of the ΣD video signals is sampled when the pilot is injected. The process is identical to the formation of the target RB values as described in section 4.3.1.

Tracking channel One I/Q pair of each of the tracking video signals: ΣT(I,Q),

ΔAZ(I,Q) and ΔEL(I,Q), is sampled for pilot purposes. The pilot is only sampled on the centre or more specifically the pilot centre gate (PCGT) for the three tracking channels. See Section 3.1 for more details.

Role of the pilot samples in normal radar operation

As mentioned above, pilot samples are needed for radar calibrations and-or RT data corrections. This section only presents the pilot samples acquired in real time, that is during a normal transmit frame. For information on the pilot signal's role in the radar calibration process see [10].

A pilot signal is injected at the end of every PRI in the receive frame. The range at which this pilot is injected is set by SEL1W13, this can be before or after the detection window. The pilot samples are then converted as explained above.

These pilot samples are used to correct gain and phase errors between the sum and the difference channels which can lead to errors in the angle measurements (azimuth and elevation). For each PRI, the complex ratio between the pilot samples in the sum channels to the difference channels is used to normalize the corresponding sum to difference ratio in the target samples.

4.3.4 Data Correction and Formatting

During a receiver channel calibration, the system computer (SEL) calculates a set of correction parameters by analysis of the pilot samples. These parameters, known as Churchill correction factors [24], are then used by the Arithmetic Calibration Card of the SVCU to do RT data corrections.

A mathematical approach

The following equations represent the RT corrections performed on the samples by the Arithmetic Card [2]:

$$\begin{aligned} \mathrm{Re}'(t) &= 8 \times (\mathrm{Re}(t) + a) \times (\frac{E+1}{2}) \\ \mathrm{Im}'(t) &= 4 \times (\mathrm{Re}(t) + a) \times P + \mathrm{Im}(t) + b \end{aligned} \tag{4.2}$$

where Re(t) and Im(t) are the real (I) and imaginary (Q) samples.

Factors 8 and 4 in the equations scale up the 12 bit A/D samples to 14 bits. This is to gain added precision from the digital multipliers in the Arithmetic Card [2]. Parameters a, b, E and P are the correction factors calculated by the system computer (SEL2W1:16). These parameters perform the following corrections [20]:

- E is the gain correction factor for the real part. It is sent as $\frac{E+1}{2}$.
- P is the gain correction factor for the imaginary part.
- a and b are the DC offset corrections.

The arithmetic unit operates on 64 samples per PRI (shown in Section 4.3.5). The output samples from the arithmetic unit are 16 bit 2s complement integers.

Data formatting

Before the digitized data is sent to the SEL it must be converted from 16 bit, 2s complement, integers to 32 bit floating point numbers. This conversion is done by a 'floating point translator array' found in the Format and Output Card in the SVCU.

The steps taken, by the Format Card, to convert the 16 bit integer to SEL 32 bit floating numbers can be found in [2].

4.3.5 High Speed Data Interface

After correction and formatting, the digitized samples, now 32 bit floating point numbers, are transferred from the SVCU to the SEL via a High Speed Data Interface (HSDI).

The HSD bus is a 32 bit parallel bus whose specified maximum rate of transfer in 0.5 MHz [20]. Practically though, this speed is fixed by the availability of data at the input and the HSDI being ready to receive data.

Target data (D/A samples) is sent to the SEL every PRI; in addition to this at the end of the frame some frame data is also sent.

Data per PRI

In a normal transmit frame the 64 data words shown below are transferred to the HSDI per PRI:

Table 4.3: Data sent per PRI from the SVCU to HSDI.

32 bit floating point numbers	**Number of samples**
Detection target sample pairs	40 words
Detection pilot sample pair	2 words
Tracking target sample pairs (ΣT_E ΣT_C ΣT_L ΔAZ_C ΔEL_C)	10 words
Tracking pilot sample pairs (ΣT_C ΣT_C^* ΔAZ_C ΔAZ_C^* ΔEL_C ΔEL_C^*)	12 words

*Samples have not had gain or phase corrections done by Arithmetic Card.

Frame data

Together with the target data in the last PRI in the frame, 14 words of status and time, and two test words are also transferred to the SEL. The two test words, known as 'the comb', are shown below. The comb indicates the end of the current frame message, and is also used for error checking.

55555555_{16} (0101...01)
$AAAAAAAA_{16}$ (1010...10)

Per transmit frame, consisting of 32 PRIs, a block of:

Target data + status and time + test words
=(64×32)+14+2
=2064

words are transferred to the SEL.

4.4 Radar Sync Pulses

See Section 3.2.2 and 3.2.3 for detailed discussions on the sync pulses the SVCU produces.

4.5 Time within the TR

The SVCU receives the current time of day, down to a resolution of 1 μs, from an external time unit (STCG). This time serves three purposes in the SVCU: (1) to start the SVCU at a known time (t_0 trigger); (2) to record the mid-frame time; and (3) to create synchronized real time clocks for the other radar subsystems. These functions are currently executed in the SVCU Time Unit Card.

4.5.1 Time, Timing and the Time Unit

The SVCU is initiated at a pre-determined and precisely measured time of day with a resolution of 1 μs. This time value was calculated in the previous signal processing (SP) cycle, SP($n-1$), by the SEL and sent to the SVCU via the MUXBUS (SEL1W14:16). The transmit frames and all other timing within the SVCU is therefore synchronized to this time.

Standard Time Code Generator

To initiate the SVCU at a specific time, the SVCU would need knowledge of what the current time is. This current time of day in provided to the SVCU by a Standard Time Code Generator (STCG).

The STCG derives its time from a RF IRIG-B time code. This time is decoded and then broadcast to the SVCU via a 37 bit parallel bus. The 37 bit bus consists of a 17 bit seconds field and a 20 bit microseconds field. These 37 bits represent the time of day after midnight or 00h00.

t_0 trigger

The leading edge of the t_0 trigger starts the SVCU, and hence the generation of the transmit frames and all timing within the radar. The t_0 trigger only occurs when a new frame request is received from the SEL, see Section 3.3.1.

The time at which the trigger is to occur is programmed, down to 1 μs, by SEL1W14:16. A comparator, made up of 37 XNOR gates, compares the current time to the programmed time. When these two times are equal, the t_0 pulse triggers and the SVCU is started.

Mid-frame time

The mid-frame time is defined as the middle of the interval between the first and the last centre measurement gates in a frame [9]. On the mid-frame time the SYNC pulse is generated. The leading edge of SYNC is used to write the current 37 bit STCG time to a set of 'real-time' latches. This time is then sent (HSDI), together with the target data, to the SEL for processing.

4.5.2 Clock Distribution

The SVCU also provides three time (IRIG) synchronized real time clocks (RTC) to the radar subsystems.

Clocks

The three real time clocks are (periods):

1. 5ms (RTC5)
2. 20ms (RTC20)
3. 1s (RTC1)

The clock pulses are active high with a 1 μs pulse width. The leading edges of these clocks are synchronized to the STCG, if present, every two seconds (rising edge of the 1s bit) to limit time drift in these clock pulses.

Two outputs are provided for each clock and their analogue properties (levels, transitions etc.) are similar to those of the ETX sync pulse.

Receiving systems

The systems that utilize these clocks are:

Pedestal controller The pedestal controller receives RTC5 and RTC20. The RTCs set the timing of the control loops that govern the pedestal motors.

Main system computer (SEL) The SEL receives RTC20 and RTC1. The SEL uses these clocks to synchronize the SP, and other tasks within the software, to the frames being output by the SVCU. This is illustrated in Figure 3.4 in Section 3.4.

Optical Tracker The optical tracker receives RTC20. It uses this clock to synchronize the rate at which it captures its images (frame rate of video).

4.6 Composite Video for A/R-scopes

Within the radar Command and Control Assembly (CCA) there are two fast, high-brightness x-y oscilloscopes [25]. These are known as the A/R-scopes. This is a hybrid of an A-scope and a R-scope, where the latter is time expanded version of the former (see Nomenclature) [3].

The A/R-scopes allow the operator of the radar to view the raw video (skin/beacon), output by the receiver, prior to any processing. This provides a RT visual feedback of the entire PRI (A-scope) and a magnified view of the detection window (R-scope). The A/R-scope enables one to search for targets and acquire them, by manually (joysticks) moving the received target echo into the detection window (required during calibrations).

Before the analogue video can be displayed though, it has to be embedded with markers. These markers show where the echo of interest (possible target) lies with respect to the various PRI events, such as the detection window. The Display Unit Card within the SVCU is tasked with embedding the raw video with these events (markers and syncs).

4.6.1 Raw Video to Composite Video

Video signals ΣV1 and ΣV2 are superimposed with trigger and marker pulses to yield the composite video signals CV1 and CV2 as shown in Figure 4.4.

Trigger and marker pulses

From Figure 4.4 the following points/markers are noted (see Section 3.2 for details on the TR frame timing):

- The start of the PRI coincides with the PTX sync.

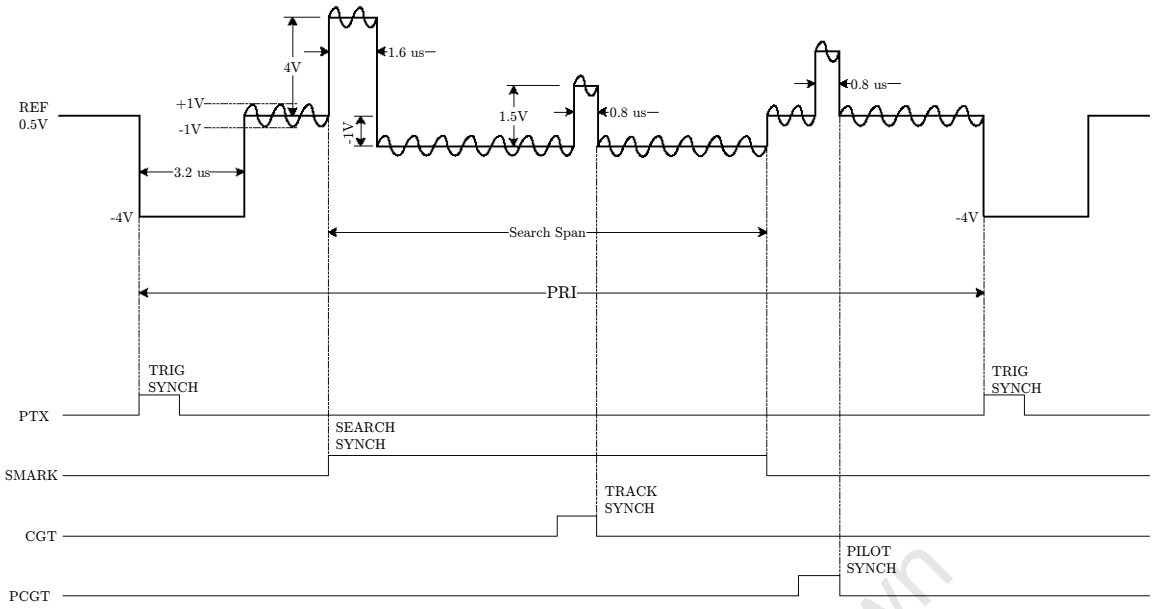


Figure 4.4: A/R-scope composite video.

- The detection window is indicated by embedding the search range start (SMARK) sync.
- The position of the tracker is shown using the trailing edge of the measurement centre gate (CGT) sync.
- The position of the pilot is indicated by embedding pilot centre gate (PCGT) sync.

All voltage levels depicted are nominal and relative to the reference level. The video amplitude is assumed to be ±1V [20].

Skin to beacon delay

The SVCU provides both skin (CV1) and beacon (CV2) video to the A/R-scopes to be displayed. CV2 only differs from CV1 in that, the start of the PRI display trigger (PTX) is delayed by the beacon delay as set by SEL1W12.

4.7 Summary

The SVCU is of a modular design, consisting of custom built cards and off-the-shelf modules. The hardware implementation of the four primary tasks of the SVCU are

summarised below.

The basebanded video signals input from the receiver are digitized in the Detection and Measurement Cards. The detection videos, ΣD(I,Q), are sampled at a constant 10 MHz. The samples are summed to form 20 range bins. The range bin values are then passed through an attenuator to make them impartial to a varying PW. The Measurement Card samples the ΣT(I,Q), ΔAZ(I,Q) and ΔEL(I,Q) video signals by means of analogue integrators. The integrated voltage levels are then corrected and digitized in a serial manner using a multiplexer.

The SVCU generates the sync pulses required by the various TR subsystems within its Event Controller Card. The properties and roles of these syncs were discussed in Chapter 3.

Time is input into the TR through the Time Unit (TU) Card in the SVCU. The TU starts the processes within the SVCU at a precisely defined time t_0. The TU also provides the mid-frame time and three time (IRIG) synchronized real time clocks (RTC) to the radar subsystems.

The SVCU also produces composite video for the A/R scopes (found on the operator's command and control assembly) by means of its Display Unit Card. Analogue videos ΣV1 and ΣV2 are superimposed with trigger and marker pulses to yield the composite video signals. These markers show where the echo of interest (possible target) lies with respect to the various PRI events, such as the detection window.

Thus far this dissertation has focussed on elucidating the TR, investigating its timing and signal processing strategies and how the SVCU executes these timing/sampling strategies. The next chapter aims to corroborate this theory by building a simulation of the TR using SystemVue™(SV) and Python.

Chapter 5

Simulation of the Lunette Tracking Radar

This chapter describes the creation of a system level model of the tracking radar (TR). This model is used to verify the theoretical and operational understanding of the SVCU; it will also serve to produce a functional specification of the hardware required to replace the current SVCU.

The simulation platforms, SystemVue™(SV) and Python, are first introduced to the reader. The overarching rules, constraints and general philosophy (approach) pertaining to the TR model is then addressed. The SV models of the various TR sub-systems are presented, these are integrated to form the complete TR model. The TR simulation is then run and the results discussed. The output from the SV TR model is then input into the Python signal processing scripts. The chapter concludes with a discussion on the output of the signal processing.

5.1 Simulation Platforms

The models of the radar transmit and receive* chain and the target echo were built using SystemVue™(SV). The output of this model was then processed by scripts, written in the Python programming language, that implement the signal processing aspects of the SEL.

*The receive chain up to and including the vector processing unit (VPU).

Figure 5.1: SystemVue and Python modelling environments.

5.1.1 SystemVue™

SystemVue™(v2010.07) is the environment chosen to build the TR and SVCU models. SystemVue™is an "electronic design automation (EDA) environment for electronic system-level (ESL) design" [6].

SystemVue™(SV) was chosen as it supports time synchronous dataflows and has an extensive RF library (filters, amplifiers, attenuators etc.). It is a GUI based environment with the ability to view signals in the time and frequency domain.

5.1.2 Python Programming Language

"Python is a popular object oriented language used for both standalone programs and scripting applications in a variety of domains. It is free, portable, powerful, and remarkably easy to use." [26]

The *Scipy* and *Numpy* Python packages (libraries) were used to re-create the various signal and data processing algorithms within the SEL (see Appendix A). These packages allow fast numeric processing and easy MATLAB®-like array handling [27]. Plots of the resulting data were done with the use of the *Matplotlib* Python package.

5.2 The TR Model Philosophy

The simulated model of the OTR tracking radar (TR) is governed by a set of constraints and characteristics. Some of these are introduced by the environment in which the simulation is done, and others are introduced to limit the complexity and scale of the simulation.

5.2.1 SystemVue^{TM}Development Environment

The characteristics and limitations introduced by the SystemVue^{TM}environment on the simulation model are discussed here.

Design type and layout

SystemVue^{TM}(SV) provides two analysis engines: (1) Data Flow - Performs a data driven analysis on data driven models; and (2) RF Design Kit Spectrasys - Performs a system-block-level non-linear analysis on a design [12]. The design of the TR model was done as a Data Flow Analysis.

Complex system designs in SV are comprised of a hierarchy of schematic, design models and parts. A part is a component from the SV library. A design model is a collection of parts that fully characterize a simulatable circuit. A schematic is then a number of interconnected design models that forms the complex system under test.

In the TR SystemVue^{TM}model, each radar sub-system is built from a set of parts to form a design model. The sub-systems are then connected together on the schematic level to form the complete TR model.

Simulation parameters

Three simulation parameters set the time domain characteristics of the SV model: (1) the start time; (2) stop time; and (3) the system sample rate (SSR). The number of samples ($n_s = \text{SSR} \times (t_{start} - t_{stop})$), time spacing ($t_{sp} = \frac{1}{\text{SSR}}$) and frequency resolution ($f_{res} = \frac{\text{SSR}}{n_s}$) are inferred from these parameters [12].

The simulation run time is proportional to the number of samples (therefore SSR) and is limited by the computer memory resources. Setting too high a SSR causes the simulation to crash due to insufficient memory allocation during setup.

To be able to run the TR SV model within a reasonable time and within the memory resources of the PC, the TR model sample rate is set at 10 MHz ($t_{sp} = 100$ ns). The timing critical signals of the SVCU are analysed separately at a SSR of 1 GHz ($t_{sp} = 1$ ns).

Component libraries

A variety of SystemVue^{TM}(SV) models/parts were used to construct the TR model. The parts stem from three SV libraries: (1) Algorithm design library for RF sources and parts, samplers, ADC models and other mathematical functions; (2) Hardware design library for boolean gates and logic functions; and (3) Miscellaneous library for input/outputs, data type conversion, capture and plotting.

The output of each SV part, in a Data Flow Analysis (discrete), is a mathematical function of the input $y = f(x)$ [12]. Most models process one input sample to produce one output sample, the exceptions to this are parts that operate on vectors and up/down-samplers.

Each model also has a set of parameters that can be tuned to set its characteristics.[1]

Data types

SystemVue^{TM}has a number of data types that can be input and manipulated by the library parts. These include scalar and matrix variables of integer (32 bit), fixed and floating (real) point, and complex type numbers.

Envelope data type In SystemVue^{TM}, a modulated passband signal is usually represented as an analytic signal [12]

$$x_a(t) = (x_i(t) + jx_q(t))e^{j2\pi f_c t} \tag{5.1}$$

in the envelope data type. The benefit of using the SV envelope signal to represent modulated signals, as compared to direct real signal representation, is that the sample rate needed to fully represent a complex envelope signal can be in the order of the information bandwidth, which is in general orders of magnitude smaller than the sample rate required for direct real signal representation [12].

The RF parts and sources in SV operate primarily on envelope signals, whereas the algorithm design library parts operate on integer and-or fixed and floating point numbers[2]. For this reason SV has a set of data converters to allow interfacing between RF and algorithmic parts (conversion rules described in [12]).

[1]For detailed descriptions of the characteristics of each SV model/part, please see [12].
[2]The input/output data type is denoted by the colour of the port as described in [12].

Data capture and plotting

Data capture in SystemVue^{TM}is done by means of a *Sink* part. This data can be stored in a data set and-or output to a text, binary or other predefined SV output file type. Data stored in a data set can be plotted in SV using the graph functions.

In the TR model data sets (graphs) were used to analyse the signal at various points in the simulation chain; data that required post processing (Python) was output to text or binary data files.

5.2.2 TR Model Constraints

The TR receive chain architecture is largely identical for the Σ, Δ azimuth and Δ elevation RF signal paths (see Figure 2.5). For this reason a decision was made to only simulate the Σ channel[3]. The Σ channel was chosen because it is used for target detection and tracking purposes (see Sections 3.5 and 3.6).

The SystemVue^{TM}Data Flow environment is time synchronous, this means all oscillators within the model are inherently synchronized to the system sample rate. This removes the need to synthesize all frequencies from a central source (for *coherency* as described in Section 3.1.2). Separate oscillators are used for LO-1, LO-2 and COHO; these are output at their optimum power levels as documented in [7].

The TR model is setup in skin mode with the transmit frequency f_c fixed at 5.5 GHz (set in RF Head model) and the TRLA and IF attenuator values also kept constant.

The radar parameters: pulse width (PW), PRI/PRF, detection and tracker range, number of transmit pulses (L) and the receiver noise level are defined as global variables that are adjustable as required.

The target range and velocity (Doppler), and the magnitude of the clutter in the scene are also defined as variables.

5.2.3 Target Scene and Noise Models

For a discrete transmitted signal $Ax_p[n]\cos(2\pi f_c n)$, the return signal is modelled as

[3]The pilot path is not modelled.

$$r[n] = A_1 x_p[n'] \cos(2\pi(f_c + f_d)n') + C[n] + N[n] \tag{5.2}$$

where A_1 is the attenuation factor inversely proportional to the square of the target model range, f_d the Doppler frequency of the target model, $n' = n - n_R$ is a shift in the discrete samples to model the delay due to the target range, and $C[n]$ and $N[n]$ the discrete Gaussian clutter and noise models.

The clutter and noise models, $C[n]$ and $N[n]$, are formed by using the SV Gaussian noise waveform generator part. The clutter signal is passed through a low pass filter to limit the bandwidth[4], whilst the noise signal is unfiltered (white noise) [3].

5.3 Signal Generation and Transmission

The transmit signal is generated within the RF Head model, shown in Figure 5.6. A PTX synchronization pulse (sync) from the SVCU model is used to switch the 120 MHz oscillator (MOSC). This 120 MHz pulse is multiplied, $\times 9$ frequency multiplier, up to 1080 MHz and then mixed with LO-1 to bring it to f_c. This signal is then filtered and amplified to a nominal peak power level of 0.025 W. This signal is fed to the transmitter model.

5.3.1 Transmitter Model

The TR transmitter consists of three amplification stages as described in Section 2.3.1. The transmitter is modelled by simply using three RF amplifiers set at the nominal gains of 16, 47 and 10 dB (see [8]) as shown in Figure 5.2. Figure 5.13 shows a spectral analysis of the transmitted pulse at a peak power level of approximately 580 kW from the model.

This signal is then routed to the antenna model via the RF Front End (RFFE) as described in Section 5.5. The antenna is modelled as a bi-directional amplifier at 43 dB (from [7]) using two amplifiers as depicted in Figure 5.11.

[4]Clutter assumed to originate from stationary and-or slow moving objects such as birds.

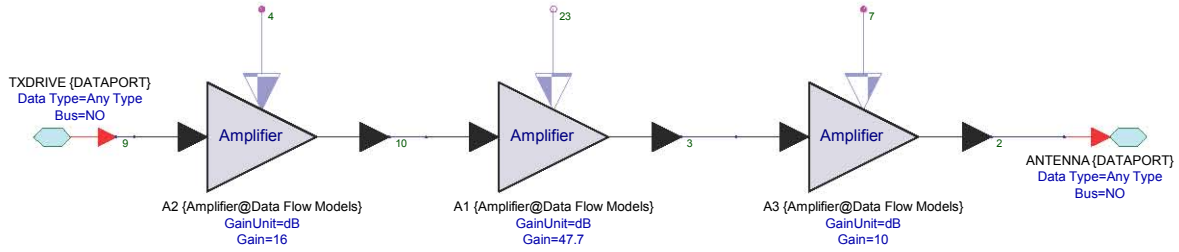


Figure 5.2: SystemVue^{TM}: Transmitter model.

5.4 Scene and Target Models

The transmitted signal is then manipulated by a target and scene (environment) model. The transmit signal is attenuated by a factor (dB) [5]

$$Atten = 10\log_{10}(\frac{\lambda^2}{(4\pi)^3 R^4}) \tag{5.3}$$

where λ and R are the wavelength and target range (one way) as defined in the model.

The attenuated signal is then mixed with a Doppler frequency f_d, and a clutter signal $C(t)$ is also added to it. The clutter is modelled as a complex Gaussian noise waveform that is low pass filtered with a cut-off frequency of 100 Hz (see Figure 5.3). A Doppler clutter band of 100 Hz encompasses sources of clutter (sea waves, birds, rain clouds etc.) with a radial velocity of up to 2.5 m s^{-1} at C band [4]. A 5^{th} order Bessel low pass filter was used.

The resulting signal plus clutter is time shifted by n_R samples, using a SV delay part, as determined by

$$n_R = \text{round}(\frac{2R}{c}\text{SSR}) \tag{5.4}$$

where $c \approx \times 10^8$ is the approximate speed of light, and SSR the system sample rate which is set at 10 MHz for the TR model simulation (Section 5.2.1). n_R is rounded to the

nearest integer as the number of samples is a $\mathbb{N}$atural number.

The target and environment model is shown is Figure 5.3. Note the use of data type converters, *EnvToCx* and *CxToEnv*, to interface envelope and complex data type parts.

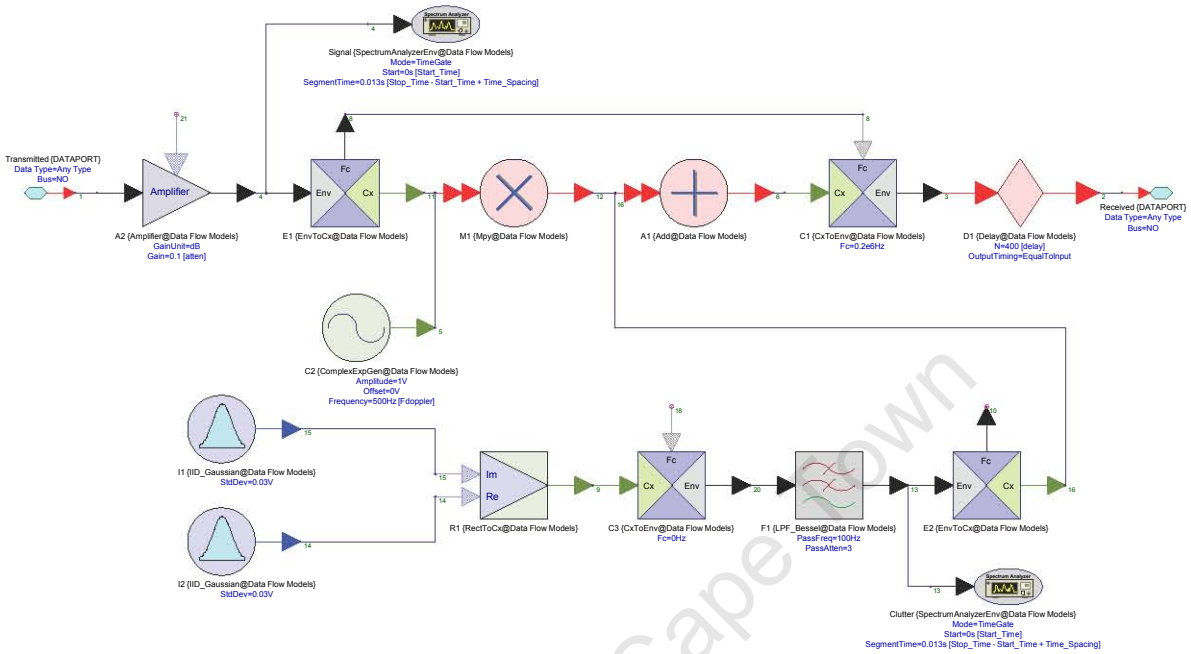


Figure 5.3: SystemVue^{TM}: Scene and target models.

5.5 Target Echo Reception and Down-conversion

5.5.1 Radar Inefficiencies

Before the target echo (transmit signal transformed by the scene and target model) is input into the receive chain, it is corrupted with noise (White) and attenuated. This is a lumped model of the receiver noise and loss down the receiver chain. This model, Figure 5.4, is aptly named the Radar inefficiencies.

5.5.2 RF Front End Model

The signal is then input into the first stage of the receive chain, the RF Front End (RFFE). The RFFE consists of a circulator, band pass filter (BPF) and low noise amplifier (LNA) connected in series.

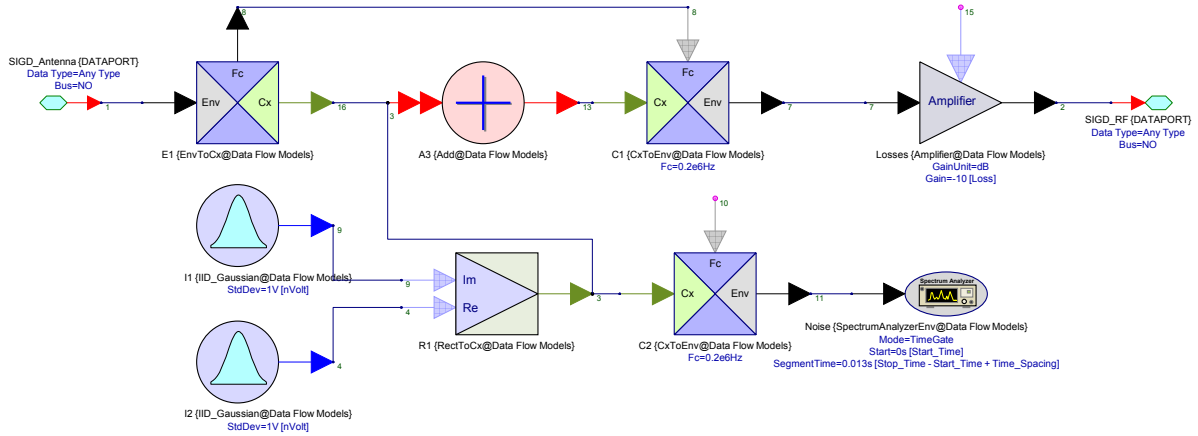


Figure 5.4: SystemVue^{TM}: Radar inefficiencies model.

The circulator is disregarded in the simulation, the transmit signal (from transmitter model) is simply connected directly to a port that is connected to the antenna model. The TRLA is modelled as a fixed attenuator. This is followed by a BPF of 5.4 - 5.9 GHz and a 30 dB amplifier (LNA). Figure 5.5 shows the RFFE model.

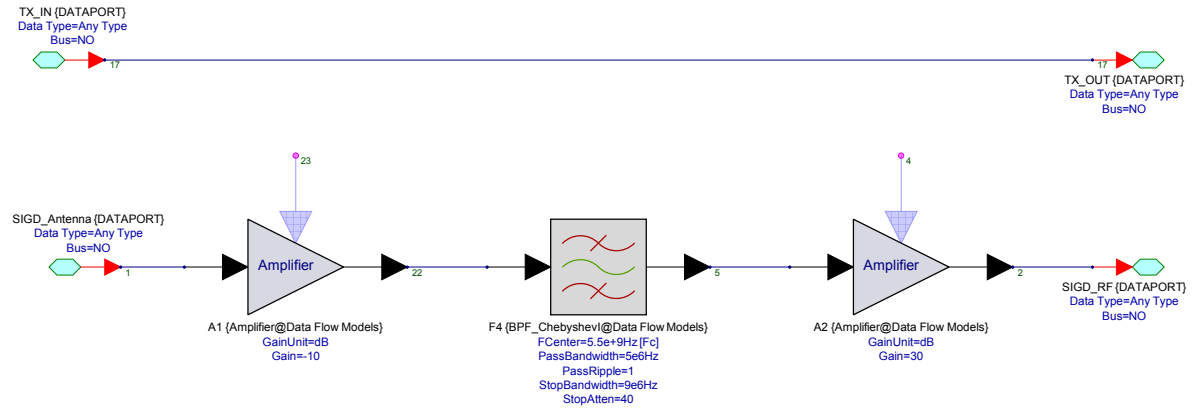


Figure 5.5: SystemVue^{TM}: RF Front End model.

5.5.3 RF Head Model

The target echo is then down-converted to an intermediate frequency (IF2 = 30 MHz) in a two stage heterodyne method. Figure 5.6 shows the local oscillators (LO), mixer and amplifier parts used to do the down-conversion. The down-converted signal is then passed through a 30 MHz BPF filter to remove any unwanted harmonics.

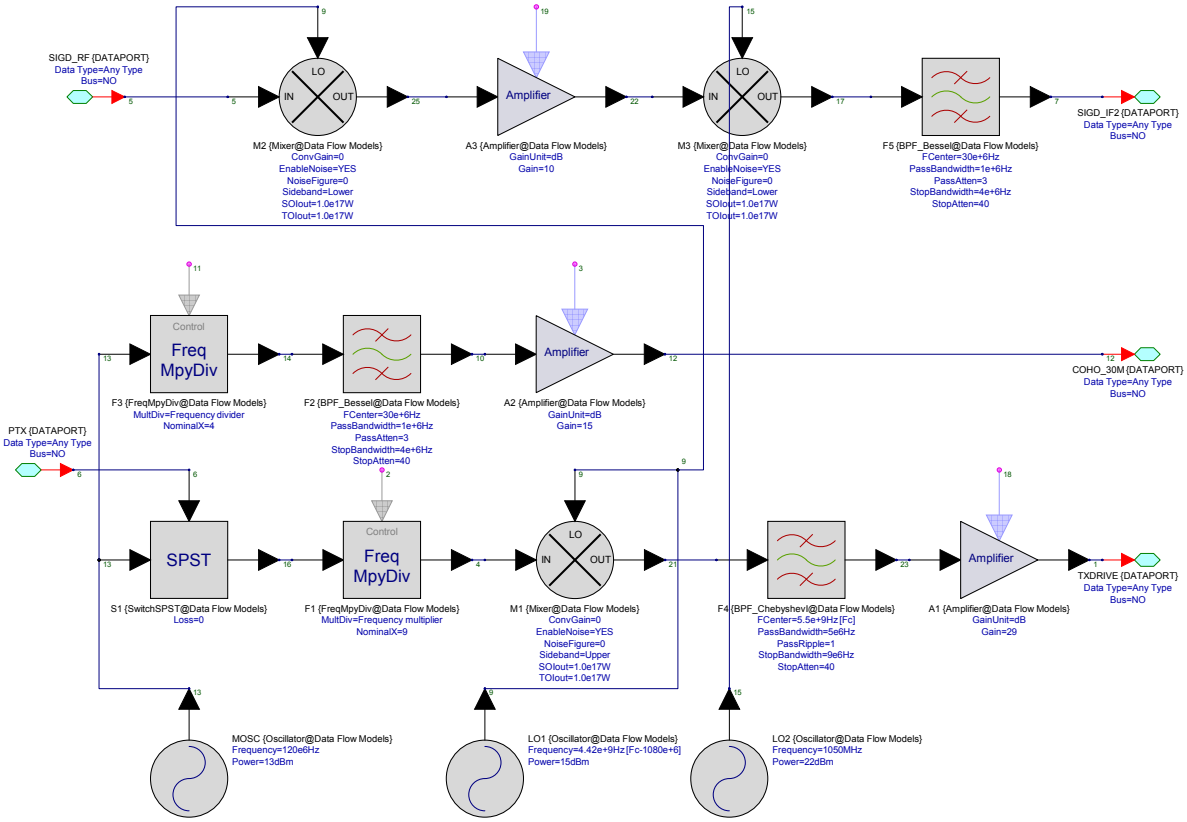


Figure 5.6: SystemVue^{TM}: RF Head model.

The COHO (30 MHz) is also generated , from the 120 MHz MOSC, in the RFH by means of a divider ($\div 4$), BPF and RF amplifier. The COHO is supplied to the TR Receiver I/Q detector.

5.5.4 Receiver Model

Only the I/Q detector function of the Receiver is modelled for the TR simulation. The phase detector is implemented by mixing the input signal with the COHO and a phase shifted (90°) COHO signal as shown in Figure 5.7.

This process results in baseband videos

$$\begin{aligned} I[n] &= A_3 x_p[n'] \cos(2\pi f_d n') + C[n] + N[n] \\ Q[n] &= A_3 x_p[n'] \sin(2\pi f_d n') + C[n] + N[n] \end{aligned} \tag{5.5}$$

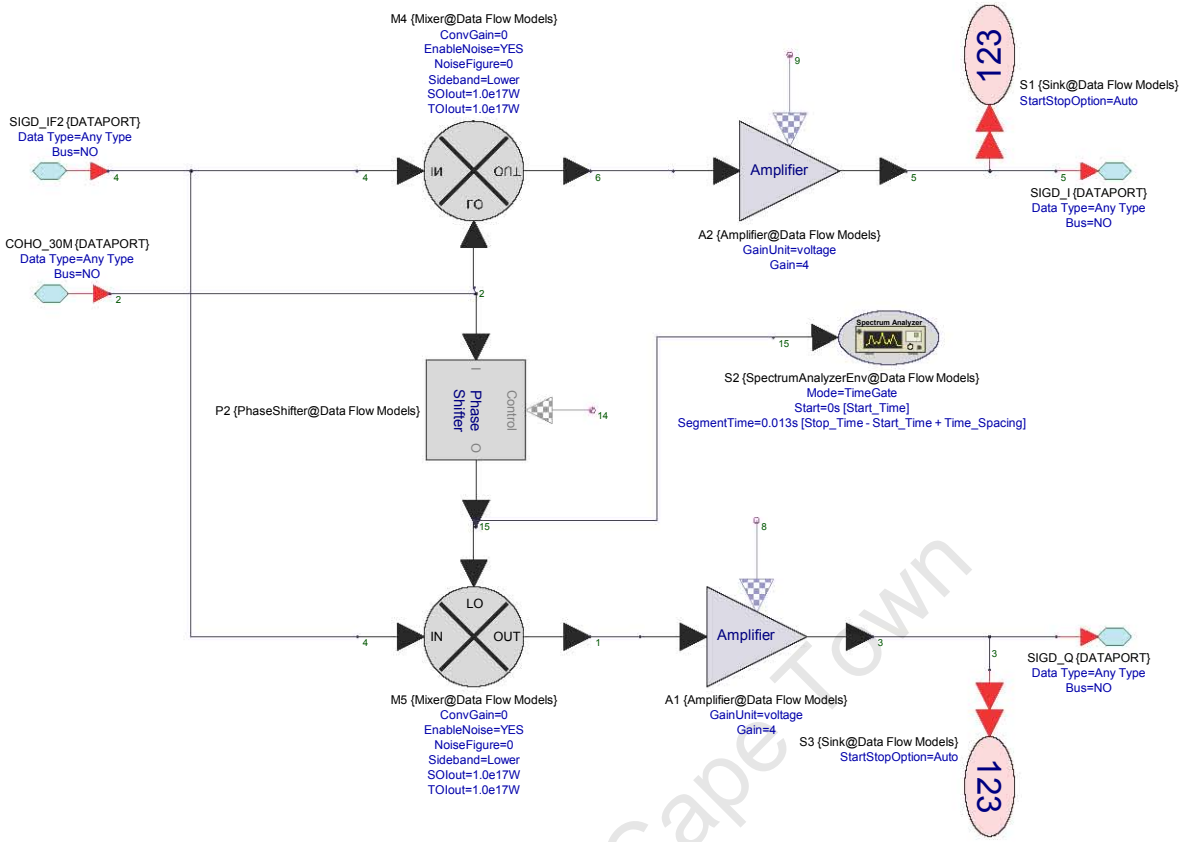


Figure 5.7: SystemVue^{TM}: Receiver model.

that are amplified to ensure it lies within the ±2.5 V level as specified in [2] and passed to the SVCU model for further processing.

5.6 SVCU Model

The SVCU model was developed in two parts: (1) the detection stream; and (2) the measurement/tracking stream. This is to allow the TR simulation, including SVCU detection channel, to run at a sample rate (SSR) of 10 MHz, whilst the more critical timing constraints of the tracking channel can be simulated at a SSR of 1 GHz.

5.6.1 Detection Channel

The signal conditioning front end (amplifier and offset circuitry) of the detection channel is lumped together as a gain block. The offset circuitry is not included because the ADC

part ($AtoD$) within SystemVue^{TM}(SV) is inherently bipolar. That is, for a specified V_{ref}, the input voltage must lie in the region $-V_{ref} \leq V_{in} \leq +V_{ref}$ [12].

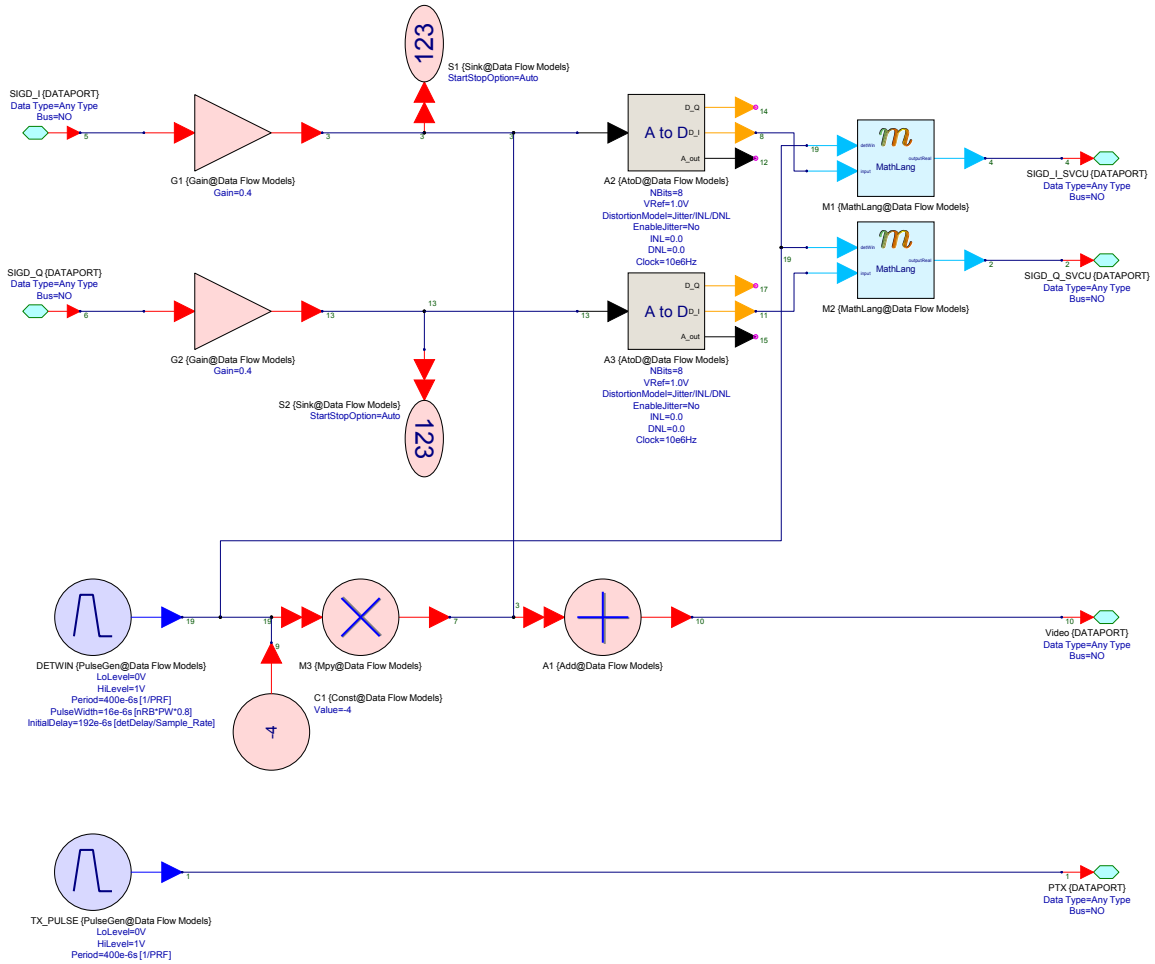


Figure 5.8: SystemVue^{TM}: SVCU detection channel, Display and Event Controller models.

Due to this restriction the $0 \rightarrow -2$ V range of the SVCU detection ADC (see Section 4.3.1) could not be simulated. The input amplifier is set such that the input signal is limited to ± 1 V. This signal is then input into the SV ADC part. The ADC is configured as an 8 bit ADC, clocked at 10 MHz (synchronized to SSR) and 2's complement output format.

The resulting digital data is then integrated (summed) to form 20 range bins, at the detection range, by means of a SV *MathLang* and a *PulseGen* part[5]. The *PulseGen* outputs the detection window sync pulse delayed by the detection range (simulation

[5]The SV *MathLang* part uses Math Language equations to process input data and produce output data [12]. The SV *PulseGen* part outputs a synchronized (to SSR) pulse with a configurable delay, pulse width and rise time.

parameter) and of width $20 \times RB$, where $RB = 0.8 \times$ PW. This pulse triggers the *MathLang* part which contains the script that forms the 20 range bin values[6].

Figure 5.8 shows that the process is identical for the I and Q channels. The resulting 20 range bin sample pairs (I/Q) are passed to the VPU for the first stage of signal processing.

5.6.2 Tracking Channel

The tracking channel model is run at a sample rate of 1 GHz and simulated independently of the main TR model (see Section 5.8.1).

The transmit sync pulse is time shifted, using a SV *Delay* part, by an amount equal to the tracker range (centre), slightly earlier (early) and later (late). This forms the three integration gates: EGT, CGT and LGT as shown in Figure 5.9. The integration gates are then inverted to form a negative going pulse (requirement for SV *Integrator* part as explained below).

The measurement gates are then up-sampled to 1 GHz, and jitter (random/periodic) and rise time pulse parameters introduced, using a SV *PulseShaping* part. The 1 GHz sample rate ($t_{sp} = 1$ ns) is to afford for the fine time measurements that the tracking channel performs. Jitter is introduced to study its effect on the range measurements (see Section 6.4).

The measurement gates drive a set of integrate and hold (I/H) circuits. The I/H circuits are modelled using the *Integrator* and *Max/Min* SV parts. The gates are connected to the reset port of the *Integrator*, which when held low integrates (sums) the input video samples(I/Q) and outputs the results [12]. The *Max/Min* part then picks the highest or lowest (negative) value per PRI.

The I/H value is then amplified (match dynamic range of ADC) and digitized by an ADC model. To simplify[†] the model, one ADC per gate sample is used.The ADC is configured as a 12 bit converter that outputs 2's complement digital data.

[6]The MathLang code listing for the range bin formation is shown in Appendix A.1.

[†]Actual SVCU uses a MUX to convert each gated sample in a pipeline method as described in Section 4.3.2

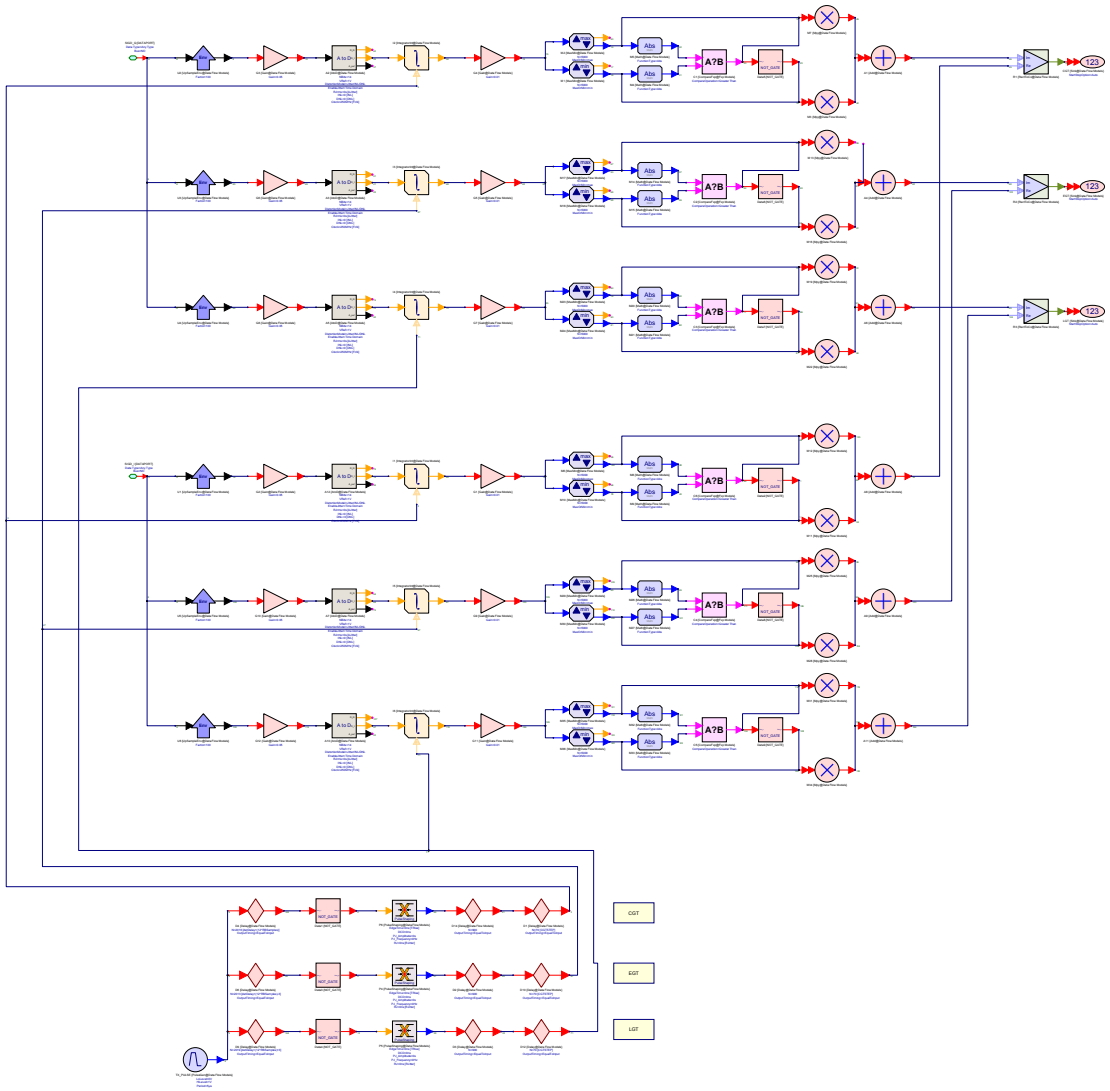


Figure 5.9: SystemVue^{TM}: SVCU tracking channel model. The model is comprised of many parts and is not clearly legible in this graphic. Please refer to the text (Section 5.6.2) for details.

5.6.3 Display Videos

The display videos for the A/R scopes (housed in TR operator's station) are embedded with a set of syncs within the SVCU Display Unit card. The characteristics of this composite video were described in Section 4.6.

This process is simulated by superimposing the detection window sync (described in Section 5.6.1) onto the baseband video using the SV *Add* and *Multiply* parts. This is shown in Figure 5.8. This composite video is then "displayed" by the A/R scope model described below.

5.7 A/R Scope Model

The A/R scopes allow the operator of the radar to view the raw video, output by the receiver, prior to any processing. This provides a RT visual feedback of the entire PRI (A-scope) and a magnified view of the detection window (R-scope).

The A/R scope plots are formed by time aligning and summing the N PRIs in a frame, given by

$$A/R[n, y] = \sum_{k=0}^{N-1} \text{PRI}[n + k \times n_{\text{PRI}}, y] \quad n = 0, ..., n_{\text{PRI}} \tag{5.6}$$

where $n_{\text{PRI}} = \frac{\text{PRI}}{t_{sp}}$ is the number of samples in the simulation PRI.

Figure 5.10 shows the SystemVue$^{\text{TM}}$A/R scope model; the model is built using a set of ten (10) SV *Sink* parts. The number of PRIs displayed was limited to 10, rather than the $N = 16, ..., 32$ PRIs transmitted, to limit the complexity of the model. The 10 datasets were then plotted using a custom graph that time aligns the data. The resulting A/R scope plots can be seen in Figure 5.16 and Figure 5.17.

5.8 Simulation Run

The sub-system models are connected together, on a schematic level, to give the TR model shown in Figure 5.11.

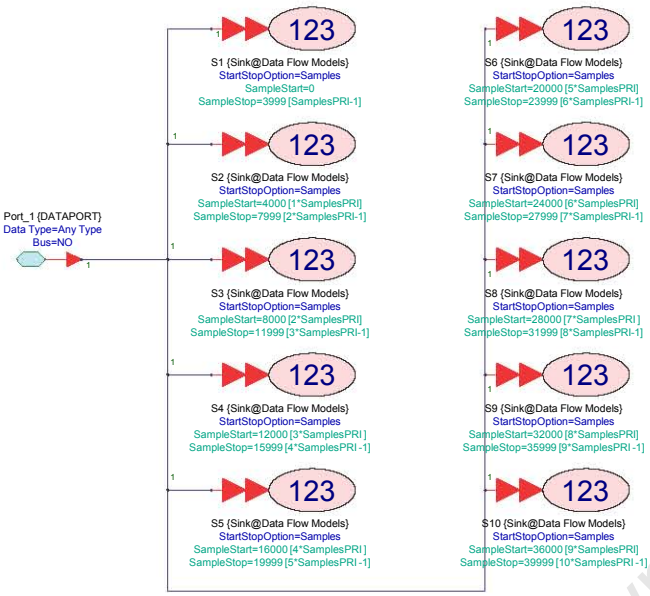


Figure 5.10: SystemVue^{TM}: A/R scope model.

Parameter	Symbol	Value
Carrier frequency	f_c	5.5 GHz
PRF	f_{PRF}	2500 Hz
Pulse width	τ	1.0 μs
Number of pulses	N	32
Detection range	R_{DETWIN}	28940 m

(a) Radar

Parameter	Symbol	Value
Range	R_{TGT}	30473 m
Doppler	f_d	684 Hz
RCS	σ	1 m^2

(b) Target

Table 5.1: Tracking Radar SystemVue^{TM}simulation set up.

The data flow is from left to right with the output data being written to a file (binary) for further processing (Python). The target range and Doppler, detection range and system noise level are set dynamically using SV *Slider* parts.

5.8.1 Set Up

The SystemVue^{TM}environment (Design Analysis) was set up with a start and stop time of 0 and 12,800 μs at a SSR of 10 MHz (128,000 samples). The TR and target models were configured as shown in Table 5.1. The noise level was set to achieve a signal to noise ratio (SNR) of 20 dB at the input to the RF Front End.

The SNR was calculated by taking the ratio of the received signal power P_r and noise power N, measured at the input to the RF Front End of the TR model. This was done

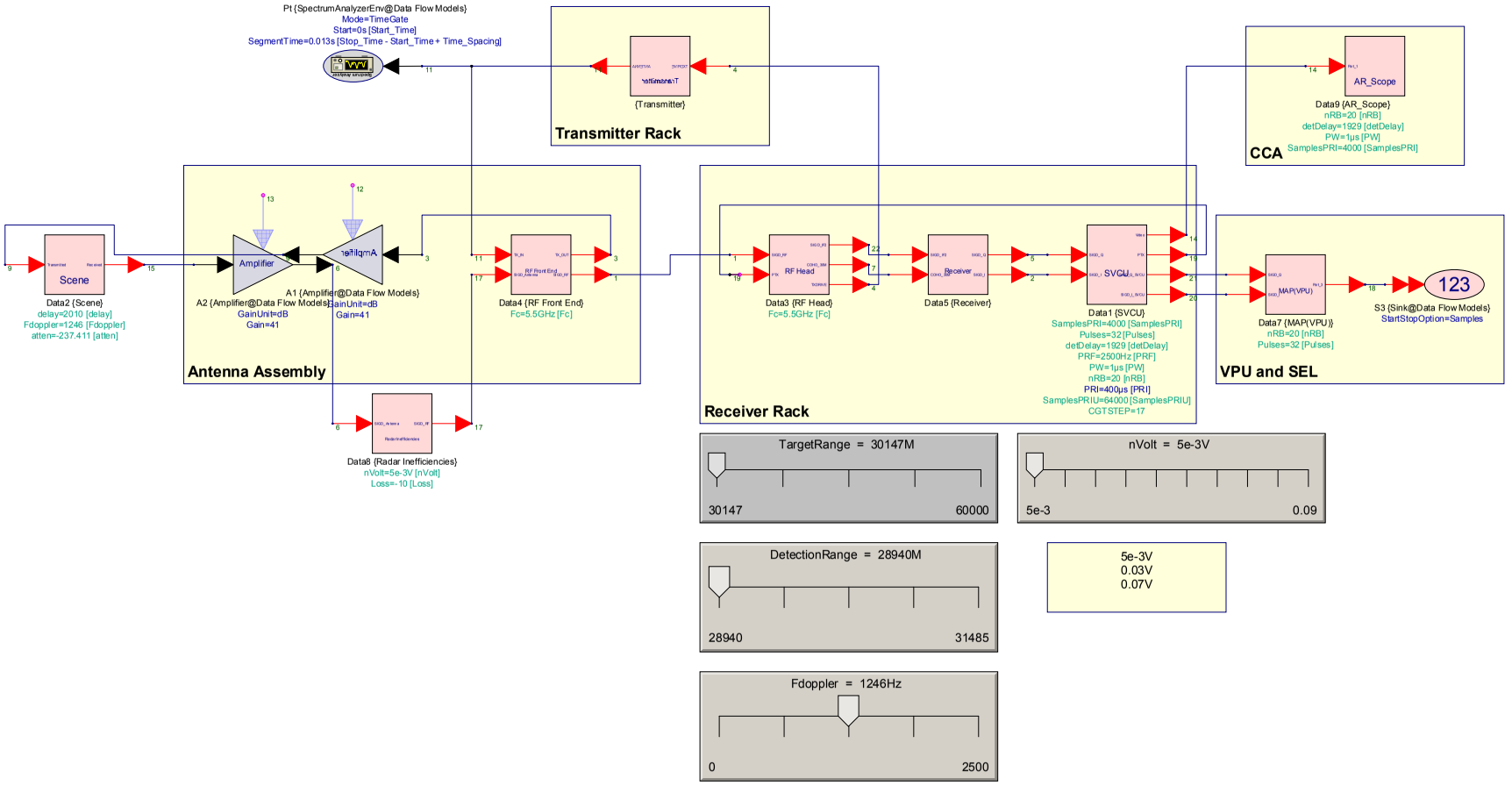


Figure 5.11: SystemVue^{TM}: Tracking Radar model.

using two SV *Spectrum Analyzer* parts[12].

The tracking channel is run, independently of the main TR model, at a sample rate of 1 GHz. Running the whole TR simulation at this rate would result in 12.8 million simulation points, which is beyond the memory resources of the PC. To achieve this, the basebanded video signals (I/Q) were recorded into a data file at the output of the Receiver (input to SVCU). A separate model was then created that comprised of just the SVCU tracking channel as shown in Figure 5.12. This model is run at 1 GHz as described in Section 5.6.2 above. The results of the tracking channel model are presented in Chapter 6.

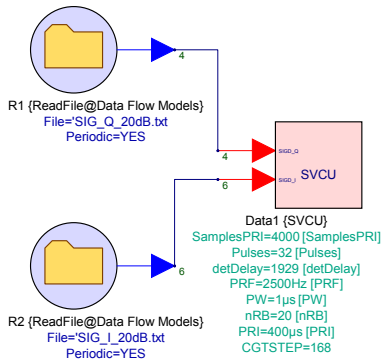


Figure 5.12: SystemVue^{TM}: TR SVCU tracking channel only model.

5.8.2 Results and Measurements

This section presents a few key plots from the SystemVue^{TM}data flow model environment.

A power spectrum analysis (SV *Spectrum Analyzer* part) of a single transmit pulse is shown in Figure 5.13. The peak power is measured at around 580 kW with a centre frequency of 5.5 GHz and first null crossings at $f_c \pm \frac{1}{\tau}$.

Figure 5.14 and Figure 5.15 show a single PRI and the entire frame (32 PRIs) of the basebanded Σ I channel video input into the SVCU.

A/R scope

The A/R scope model plot for the TR receive signal is shown in Figure 5.16. The top plot is the A-scope which shows the full PRI with detection window lying in the middle of the

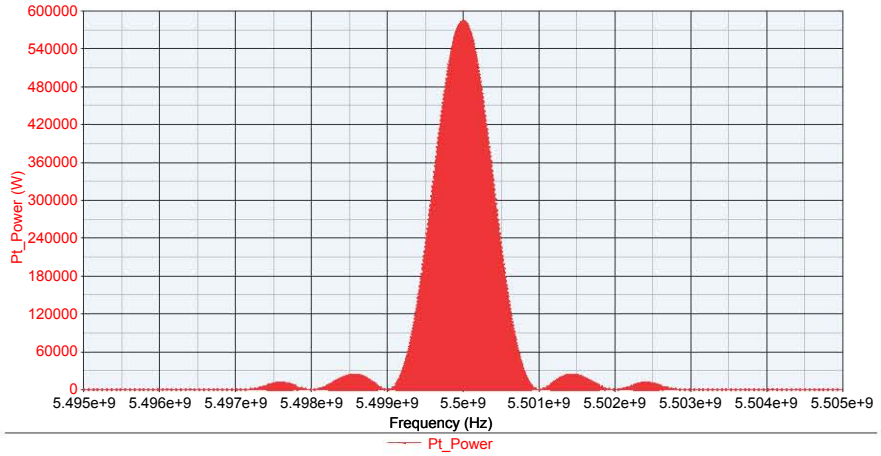


Figure 5.13: SV Plot: TR transmit pulse at output of transmitter model.

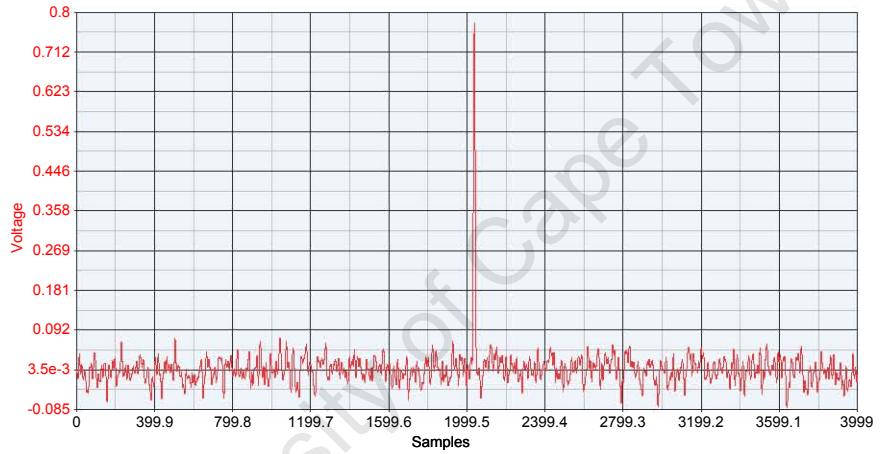


Figure 5.14: SV Plot: One PRI of TR receive signal at input to SVCU.

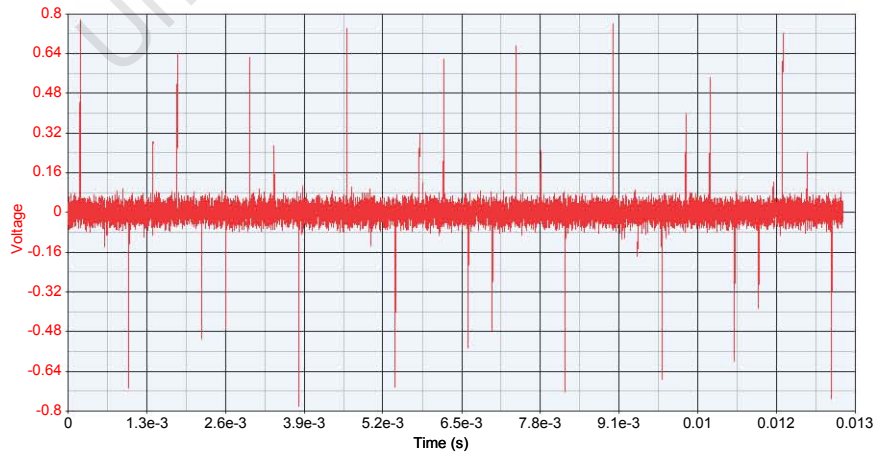


Figure 5.15: SV Plot: Receive frame (32 PRIs) of TR receive signal at input to SVCU.

PRI. The bottom plot, R-scope, shows a time expanded view of the detection window or search span.

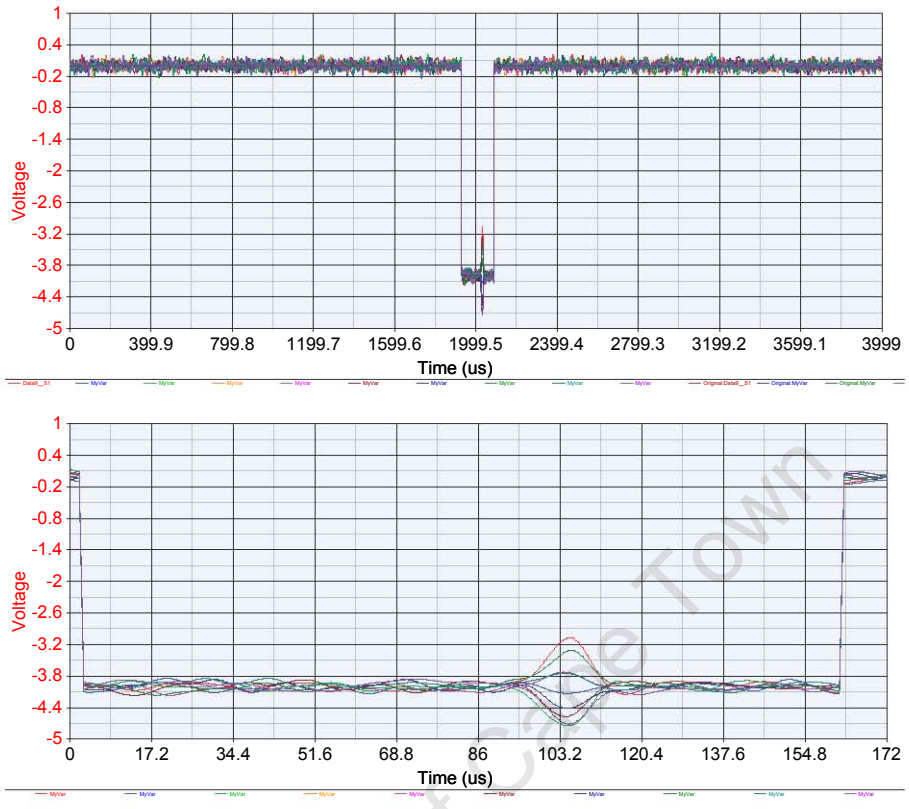


Figure 5.16: SV Plot: A/R scope showing full PRI (top) and zoomed in detection window.

The shape of the target response (shown in detection window), commonly referred to as the 'onion' or 'butterfly', is due to the Doppler (velocity) of the target model [4]. Figure 5.17 shows what the detection window (R-scope) would look like for a stationary target ($f_d = 0$). Here all the received target echoes are in phase as expected from Equation 5.5.

5.9 Signal Processing

The digitized samples from the SVCU are then input into the VPU model and signal processing scripts.

5.9.1 Vector Processing Unit Model

At the core of the VPU model is a SV *MathLang* part that contains the FFT algorithms. The incoming samples (I/Q pairs) are first combined to form complex numbers. These

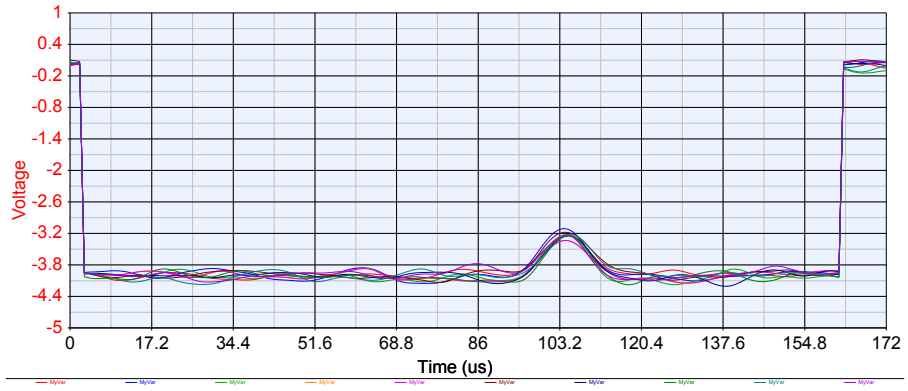


Figure 5.17: SV Plot: R-scope displaying received signal from a stationary target.

complex numbers are then input into the *MathLang* part for processing. The *MathLang* part implements the *coherent* integration as per Equation 3.11†.

A log of the samples, prior to the VPU, is also made to use for *non-coherent* processing (spectral analysis not required).

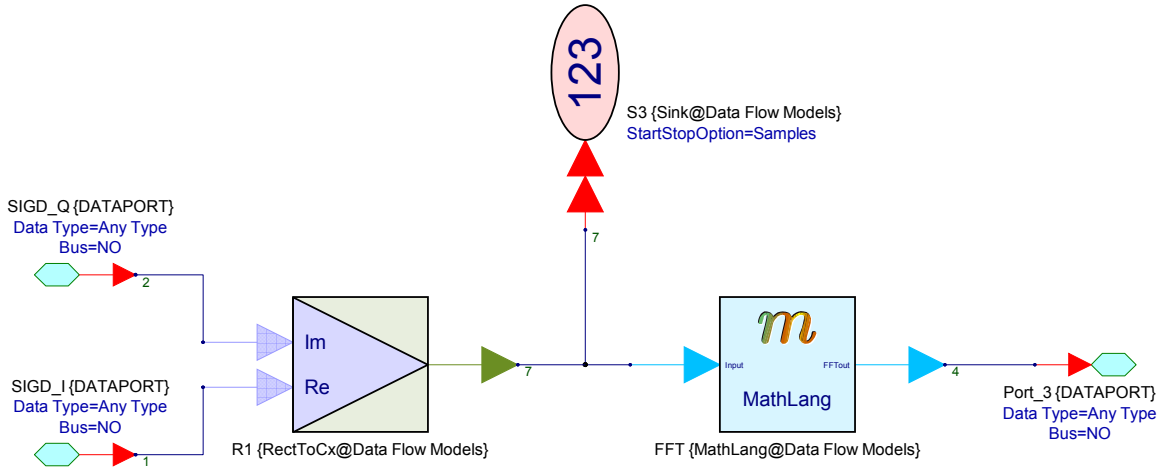


Figure 5.18: SystemVue^{TM}: VPU model.

The VPU processed and unprocessed data are then saved to binary data files for further signal processing.

†The MathLang code listing is found in Appendix A.2 and is only pertaining to detection stream.

5.9.2 SEL Model (Python Scripting)

The *coherent* and *non-coherent* binary files are then read into two 20×32 (row×col) element arrays in the Python environment. These constitute the 20 range bin samples from 32 PRIs.

Non-coherent detection

In the *non-coherent* case (data unprocessed by VPU) the array is collapsed laterally. That is, the absolute value of the samples belonging to each range gate is summed as given by Equation 3.16. The result is a 20 element vector containing the power of the signal in each of the range gates. A thresholding is then applied on this vector to determine which range bin (RB) the target lies in.

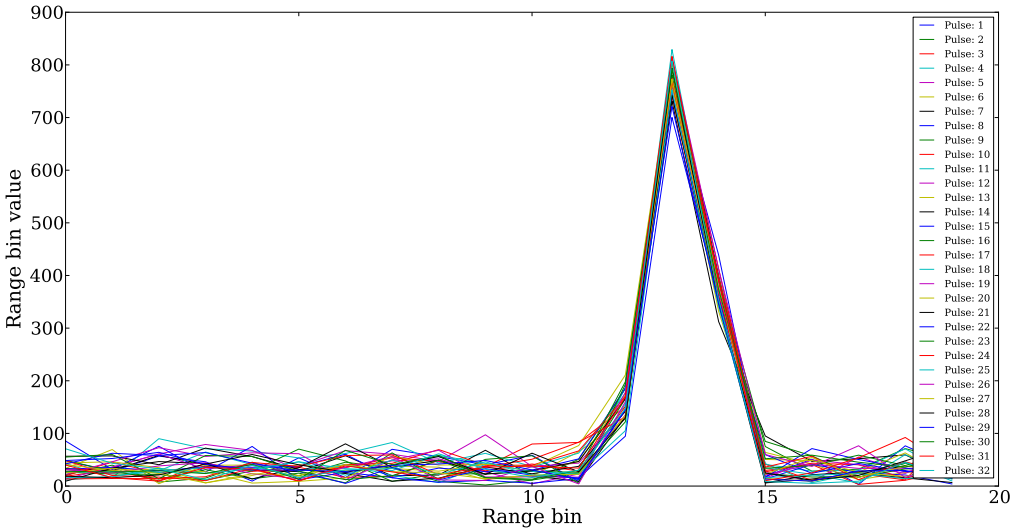


Figure 5.19: Python: 20 range gates of 32 PRIs in radar receive frame.

Figure 5.19 shows the 20 range gates for all 32 pulses (PRIs), and Figure 5.20 shows the resulting vector when the PRIs are integrated *non-coherently*. The thresholding reveals the target to be lying in the 13^{th} range bin ($m_{tgt} = 13$).

Coherent detection

The data output by the VPU has already been *coherently* integrated, forming a range-Doppler map. Figure 5.21 shows a plot of this data (20×32 element array), using the Python *Matplotlib* package, in a 3 dimensional (3D) and intensity plot.

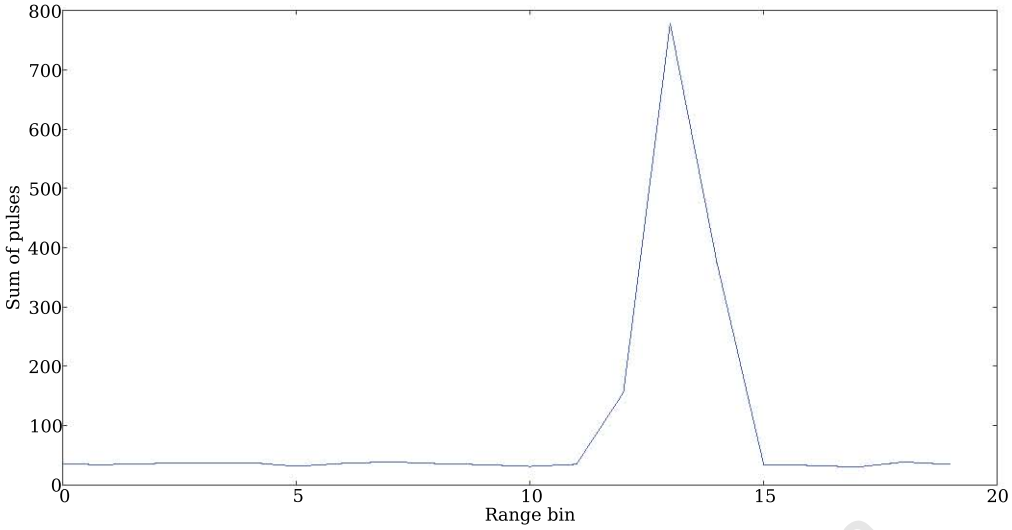


Figure 5.20: Python: Non-coherently integrated pulses.

A threshold detector is applied on the range-Doppler map and identifies the target as lying in the 13^{th} range gate and 9^{th} Doppler bin ($m_{tgt} = 13$, $k_{tgt} = 9$).

Coarse target range and Doppler measurement

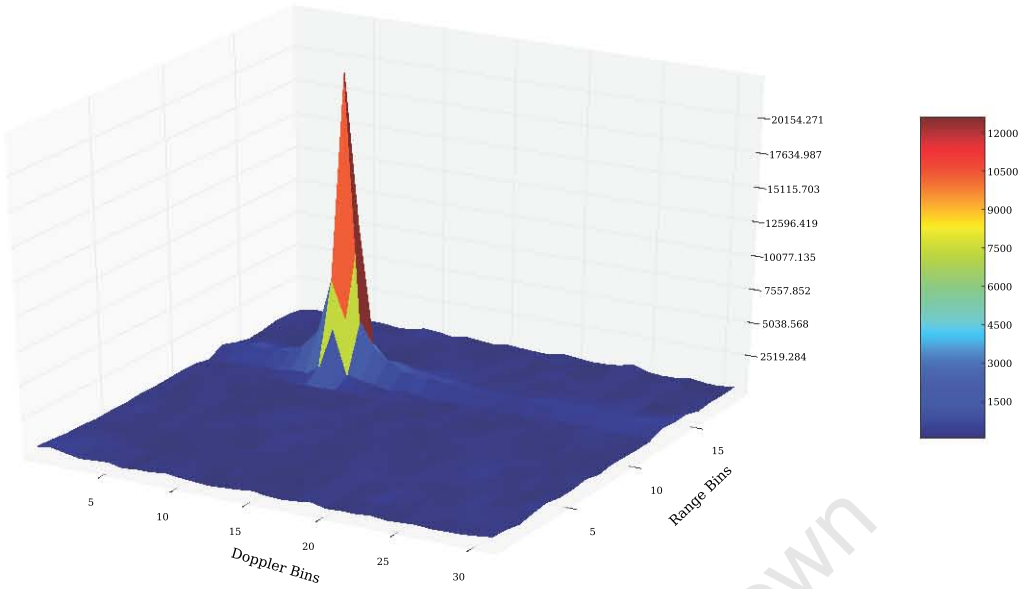
The coarse ambiguous range and Doppler for the detected target is measured as (from Equation 3.20)

$$\begin{aligned} R_{det} &= R_{DETWIN} + (m_{tgt})(0.8 \times \tau)(\frac{c}{2}) \\ &= 28940 + (13)(0.8 \times 1.0 \times 10^{-6})(\frac{3 \times 10^8}{2}) \\ R_{det} &= 30500 \text{ m} \end{aligned}$$

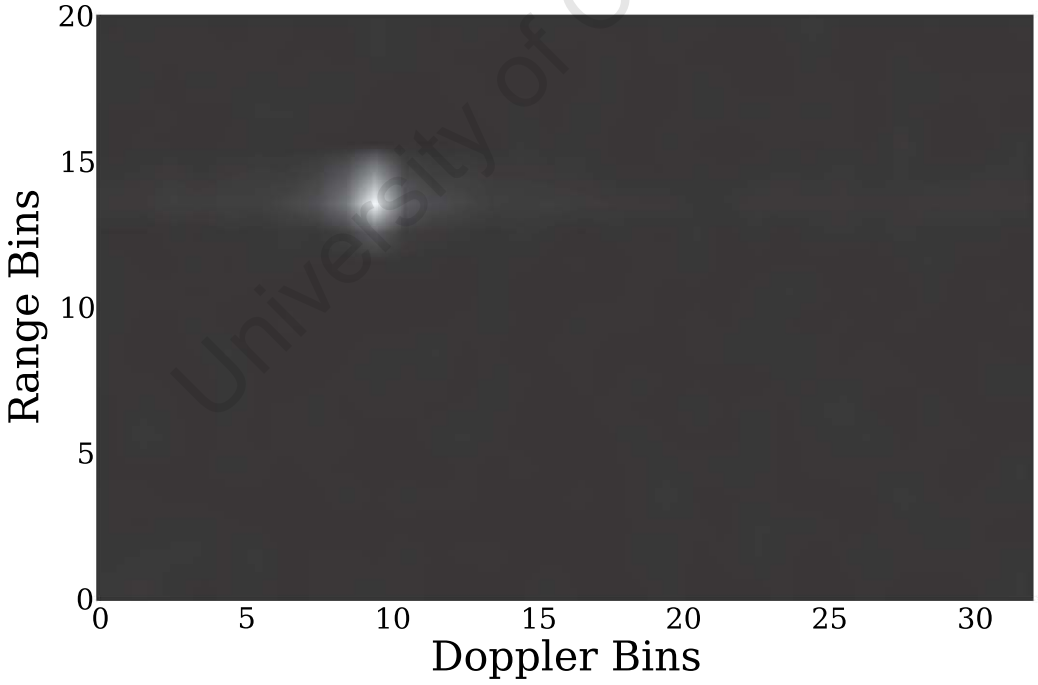
$$f_{d_{det}} = \frac{k_{tgt}}{N}(f_{\text{PRF}}) = \frac{9}{32}2500 = 703 \text{ Hz}$$

where R_{DETWIN} is the range to the start of the detection window (search span) and N the number of pulses in the PRI, and hence points in the FFT, from Table 5.1.

The measured target range and Doppler are within one range-Doppler cell resolution



(a) 3D range-Doppler map



(b) 2D range-Doppler intensity plot

Figure 5.21: Python: Coherently integrated pulses.

$$
\begin{aligned}
R_{res} &= (0.8 \times \tau)(\frac{c}{2}) = 120 \text{ m} \\
f_{d_{res}} &= \frac{f_{\text{PRF}}}{N} = 78 \text{ Hz}
\end{aligned}
\tag{5.7}
$$

of the simulated target parameters (Table 5.1).

5.10 Summary

Before setting out to build a radar model, the overarching rules, constraints (imposed by the simulation environment, self-imposed or other) and general philosophy (approach) pertaining to the TR model was addressed. The SystemVue$^{\text{TM}}$(SV) models of the various TR sub-systems were then constructed and integrated to form the complete TR model.

The TR simulation was run with user defined system (TR) and target properties. Analysis of the RF signal at the output of the transmitter chain, input into the SVCU (basebanded I/Q video) and A/R scope plots, corresponded to actual signals captured at the TR. The output of the SVCU model was then processed by FFT algorithms written in SV's *MathLang* language. These values were then input into the Python signal processing scripts.

A *coherent* and *non-coherent* thresholding detector was then applied and identified the simulated target correctly on the range-Doppler plane. The measured target range and Doppler were within one range-Doppler cell resolution of the actual simulated target parameters.

To measure the target range and Doppler to a finer resolution, the SVCU tracking channel and accompanying measurement gates are used as described in Section 3.6. The measurement of range and Doppler, using the measurement gates (EGT, CGT and LGT), is thus the most timing critical aspect of the SVCU. This process (tracking channel) and the effects of impurities (offsets, rise time and jitter) on the measurement gates on the radar measurements are investigated in the next chapter. This investigation will inform the decision as to what specifications the hardware that will shape the new TR synchronizer (SVCU) must achieve.

Chapter 6

Characterisation of Critical Timing Signals in the SVCU

This chapter describes the measurement/tracking operation of the TR and investigates the timing critical signals within the SVCU that drive this process. The setup of the SystemVue™(SV) tracking channel model is first described, followed by the associated signal processing implemented in the VPU. The range and Doppler error models of the TR are then developed. The tracking channel model is then run, yielding an accurate measurement of the target parameters to within 1 SI unit.

The tracking gates are then corrupted with jitter, rise time and offsets, the effects of this on the radar measurements is documented. A new SVCU architecture is then proposed, using a fast ADC upfront and performing the tracking gate integration digitally. The effects of aperture jitter and differential and integral non-linearities on this model is analysed. The role of the transmit sync (TX) in the measurement process is then examined and the jitter budget of the TR is set.

The chapter concludes with a case study to determine the jitter budget achievable in an FPGA based SVCU design.

6.1 Tracking Channel Model

The SystemVue™(SV) tracking channel model, and the main components thereof, have already been described in Section 5.6.2 and shown in Figure 5.9. This model is run at a system sample rate (SSR) of 1 GHz. The early, centre and late tracking gates are set

to sample the ΣT signal around the detected target range, R_{det}, as determined from the detection model simulation (Section 5.9.2). The digitized samples are output to a binary file for further processing.

6.1.1 VPU and SEL Model (Python Scripting)

The tracking data is then read into a 3×32 (row×col) element array in the Python environment. This constitutes the 3 tracking samples (early, centre and late) from the 32 PRIs in the radar frame.

Coherent measurement

Range The early and late gate samples are summed *coherently* by means of a DFT as per Equation 3.22. The error between the estimated (from detection model run) and actual target range is then calculated as

$$\varepsilon_R = K_R \left[\text{real} \left(\frac{E - L}{E + L} \right) \right] \tag{6.1}$$

where E and L are the *coherent* summation of the 32 early and late gate samples, and K_R is a factor to convert the electrical error to a physical range error. K_R is calculated by building a range error model of the TR, this process is described in Section 6.2.1 below.

Doppler A DFT of the centre gate sample is done at a frequency slightly above and slightly below the detected Doppler frequency ($f_{d_{det}}$) as shown in Equation 3.24. These two values are then used to calculate the error between the estimated and actual target Doppler frequency, given as

$$\varepsilon_D = K_D \left[\text{real} \left(\frac{Y(f_{d_{det}} + \Delta f) - Y(f_{d_{det}} - \Delta f)}{Y(f_{d_{det}} + \Delta f) + Y(f_{d_{det}} - \Delta f)} \right) \right] \tag{6.2}$$

K_D is a factor to convert the electrical error to a physical frequency (Doppler) error. K_D is calculated by building a Doppler error model of the TR, this is described in Section 6.2.2 below.

Non-coherent measurement

In *non-coherent* processing only the target range is measured. The amplitudes of the 32 early and late gate samples are summed as per Equation 3.25. The error, between the estimated and actual target range, is then calculated as

$$\varepsilon_R = K_R \left(\frac{E - L}{E + L} \right) \tag{6.3}$$

6.2 TR Error Models

The TR errors models were developed to convert the electrical signal errors to a physical range or Doppler error.

6.2.1 Range Error Model

The TR range error model, K_R, is determined by introducing a known range error and measuring the corresponding electrical error. This was implemented by sweeping the early and late tracking gates across a target at a known range R_0 as shown in Figure 6.1.

Figure 6.2 shows the early and late gate magnitudes, and Figure 6.3 the corresponding range error model, when such a sweep was done on the SystemVue^{TM}TR model (Python plots).

6.2.2 Doppler Error Model

The Doppler error model, K_D, was calculated in a similar fashion as the range error model. For an echo containing a known Doppler frequency f_d, a set of DFTs are done for a range of frequencies around the target Doppler. The corresponding electrical errors are measured. This method is illustrated in Figure 6.4.

Figure 6.5 shows the result of a set of DFTs around a given target Doppler frequency in the SV simulation. The TR Doppler error model of the simulation is then formed as per Figure 6.6.

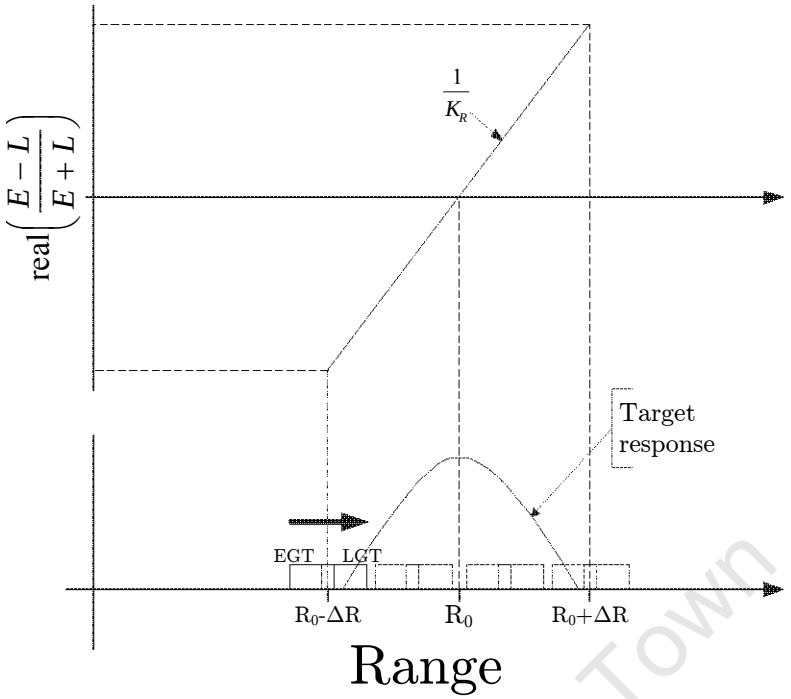


Figure 6.1: Method used to build the range error model.

6.3 Simulation Run

The TR tracking channel model is run at a SSR of 1 GHz. From the detection process (see Section 5.9.2): the centre tracking gate is placed at a range of 30500 m and the DFTs are performed, in the Python signal processing scripts, at an assumed Doppler frequency of 703 Hz.

6.3.1 Range and Doppler Measurements

A difference-to-sum ratio of the early and late, and higher and lower frequency DFTs are then calculated. These values are then interpolated onto the TR range and Doppler error models obtained from the 'calibration' process (sweeps).

The output of the tracking signal processing in shown in Figure 6.7a. The detection estimates produce range and Doppler errors of $\varepsilon_R = 19.75$ m and $\varepsilon_D = 18.96$ Hz.

Combining the initial estimate (detection) with the output of the tracking process, the target range and Doppler are measured as

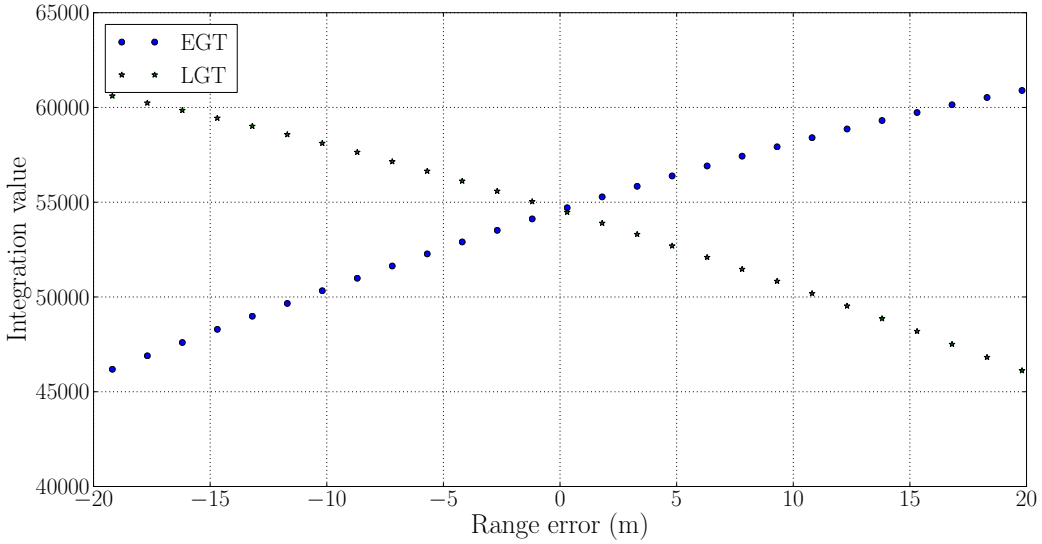


Figure 6.2: Python: Sweep of early and late gate across stationary target.

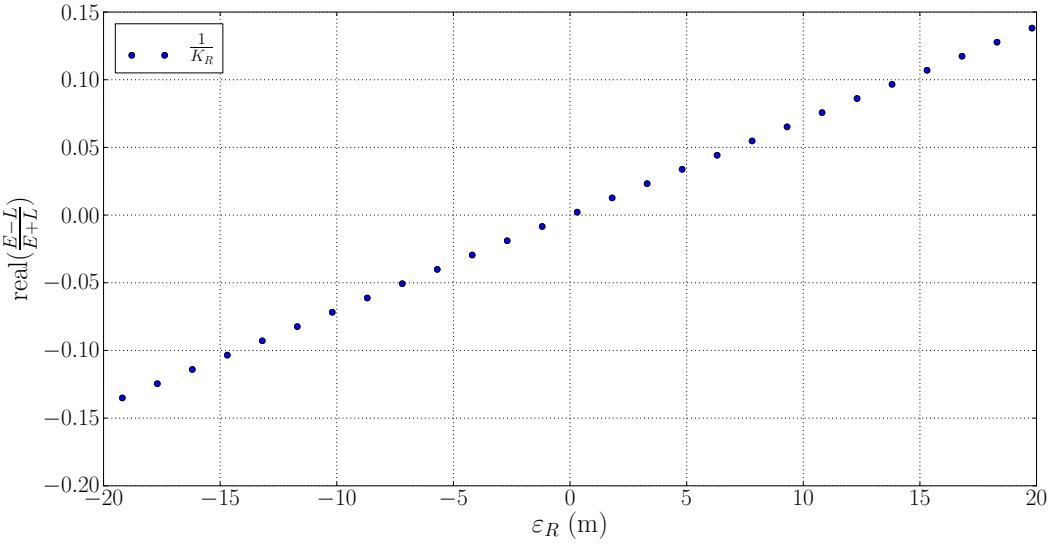


Figure 6.3: Python: TR range error model.

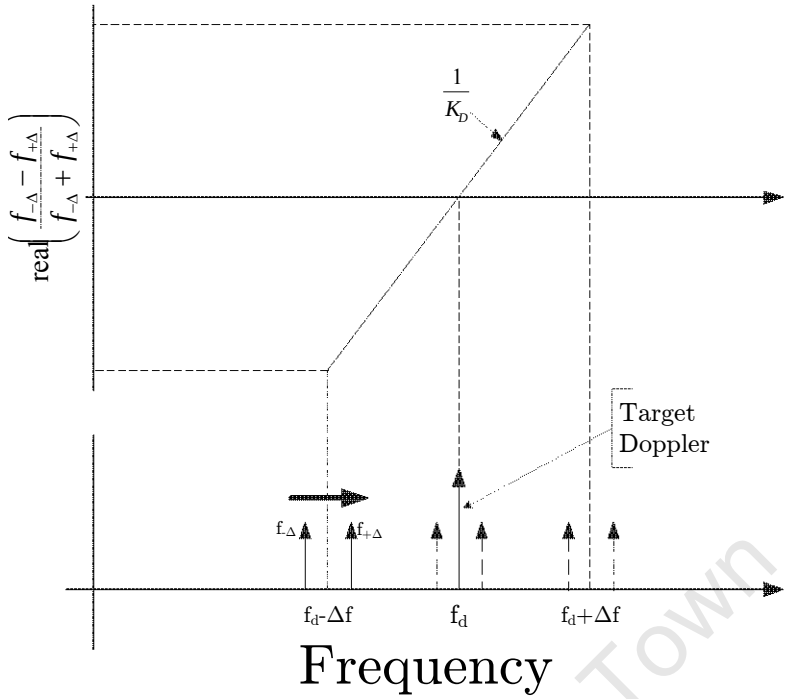


Figure 6.4: Method of building the Doppler error model.

$$R_{track} = R_{det} - \varepsilon_R = 30480.24 \text{ m}$$
$$f_{d_{track}} = f_{d_{det}} - \varepsilon_D = 684.04 \text{ Hz}$$

The tracking channel model is able to measure the target range and Doppler to within 1 SI unit of the actual target model parameters (Table 5.1).

The tracking process, within the TR, strives to minimize ε_R and ε_D by centring the target within the range and Doppler tracking gates. To simulate this, the SV model is rerun with the tracking gates placed at R_{track}. The signal processing scripts then evaluates, by means of a DFT, this data around $f_{d_{track}}$. The errors output from the centred estimates are shown in Figure 6.7b.

6.4 Corruption of Synchronization Pulses

The measurement gates (EGT, CGT and LGT) produced by the SVCU are digital pulses in nature. The hardware used to produce these pulses introduce timing constraints such

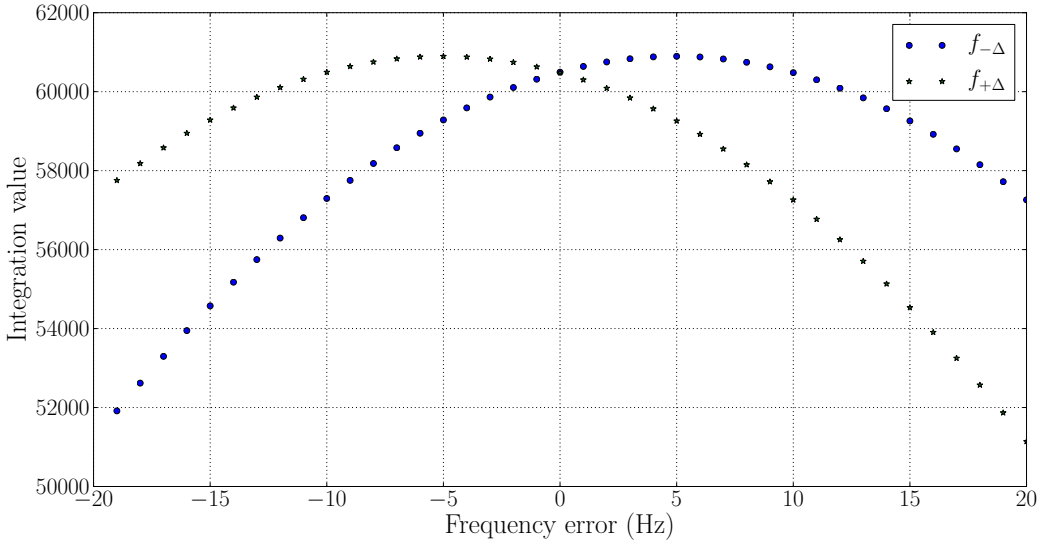


Figure 6.5: Python: Analysis of $f_{-\Delta}$ and $f_{+\Delta}$ around target Doppler f_d.

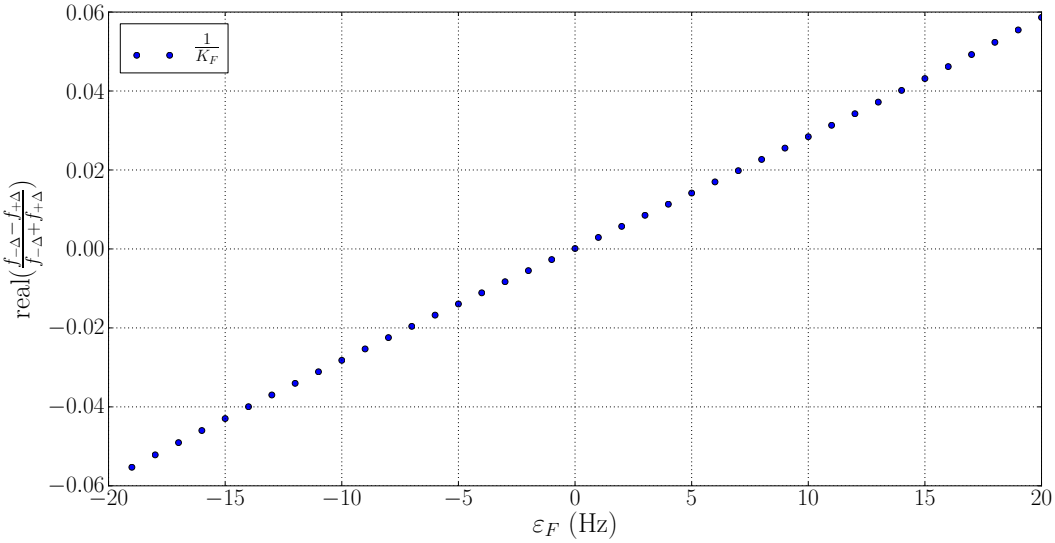
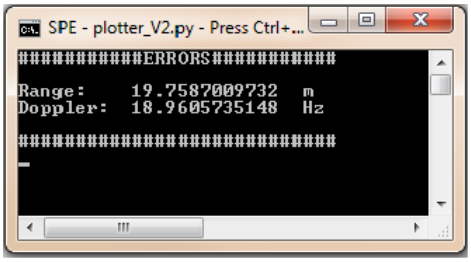
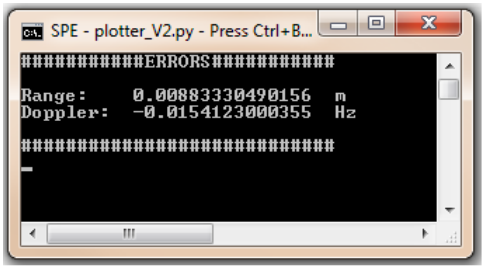


Figure 6.6: Python: TR Doppler error model.



(a) Detection: $R = 30500$ m, $f_D = 703$ Hz



(b) Centred: $R = 30480$ m, $f_D = 684$ Hz

Figure 6.7: Output of tracking channel model.

as rise time and jitter. Since the purpose of the sync pulses is measurement (range and Doppler), to a very high degree of precision, the effect of these timing constraints on the radar measurements are critical.

The tracking gates are first placed on target, i.e. such that (from Figure 6.7b):

$$\begin{aligned} \varepsilon_R = &\quad 0.008 \quad \approx 0 \text{ m} \\ \varepsilon_D = &\quad -0.014 \quad \approx 0 \text{ Hz} \end{aligned} \tag{6.4}$$

Jitter and rise time are then introduced, the resulting degradation of the range and Doppler accuracy is then measured.

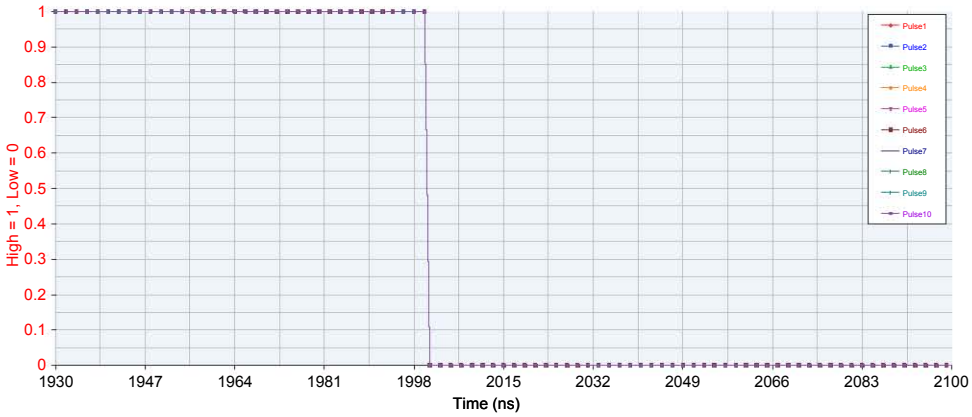
6.4.1 Jitter

Jitter in this context is defined as the random (noise-like) fluctuations of the sync pulse position. Jitter is introduced into the SystemVue™model using a *PulseShaping* part. The jitter is specified as a time domain RMS value [12].

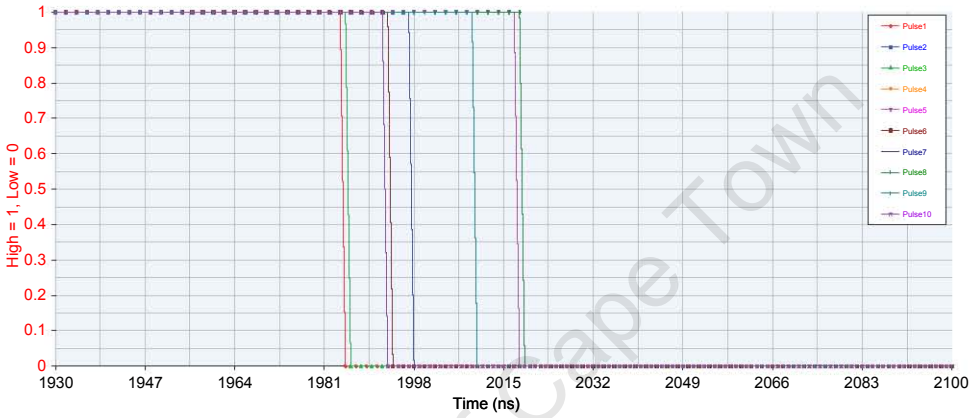
Figure 6.8 shows the effect of 10 ns of jitter on the leading edge of the centre gate (negative going pulse) of ten consecutive PRIs.

The three tracking gates are corrupted with a zero-mean Gaussian jitter. The range and Doppler errors are then observed for 100 frames (100×32 PRIs). A histogram of the resulting 100 range and 100 Doppler measurements is then plotted.

Figure 6.9 shows the results for the range measurements when there is no jitter on the measurement gates. As expected all 100 range measurements result in $\varepsilon_R = 0.008 \approx 0$ m. The dotted line represents a Gaussian fit of $\mu \pm 3\sigma$ of the 100 measurements. Assuming



(a) Jitter = 0 ns



(b) Jitter = 10 ns

Figure 6.8: SV Plot: Jitter introduced using *PulseShaping* Part.

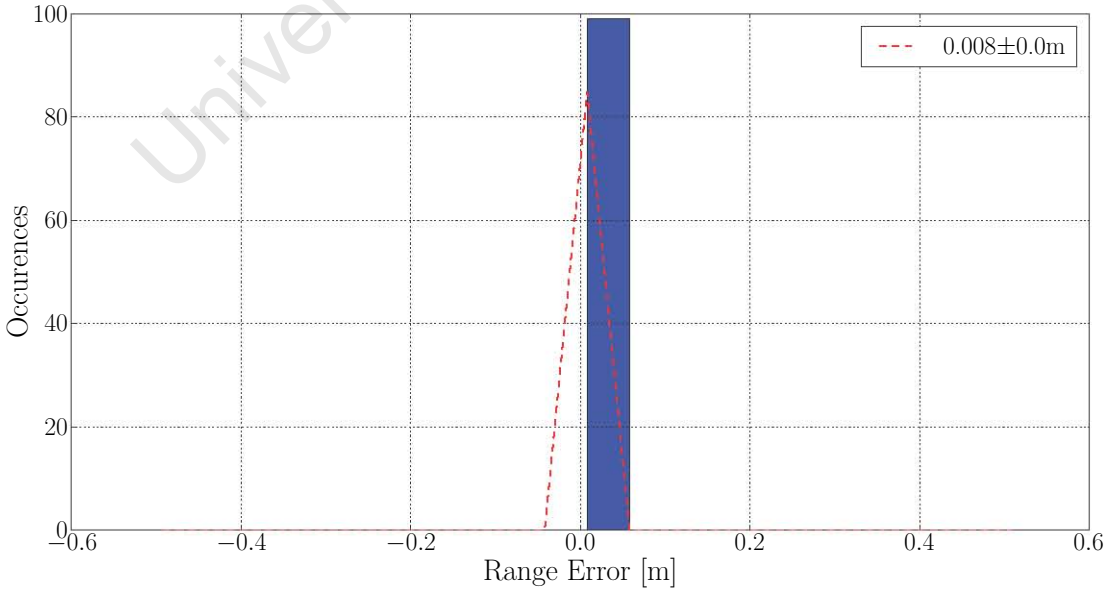


Figure 6.9: Python: Measurement gates with no jitter present.

$\mu \approx 0$ (zero-mean jitter), the 3σ value will be used as a figure of merit to evaluate the integrity of the measurement (range or Doppler).

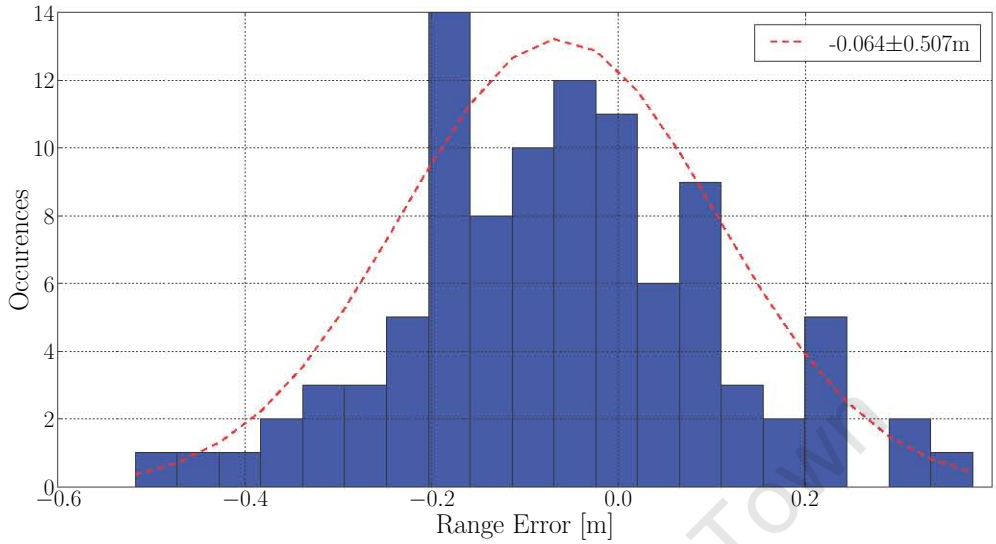


Figure 6.10: Python: Ten (10) nanoseconds jitter introduced into measurement gates.

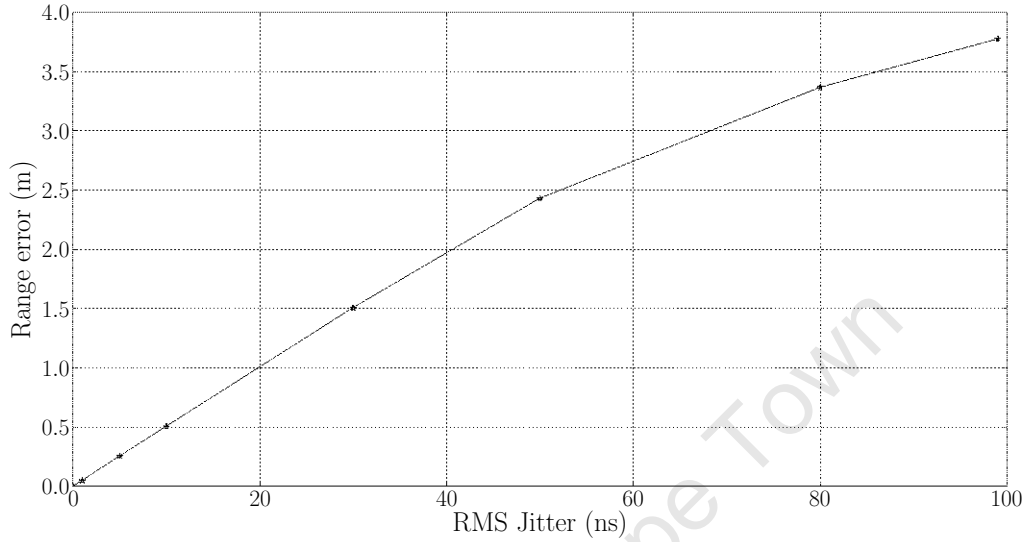
In Figure 6.10 a jitter of 10 ns is introduced onto the three tracking gates. The range measurement error (3σ) is now half a metre (0.507 m). This process is then repeated, for increasing jitter, to produce Figure 6.11.

6.4.2 Rise Time and Offset

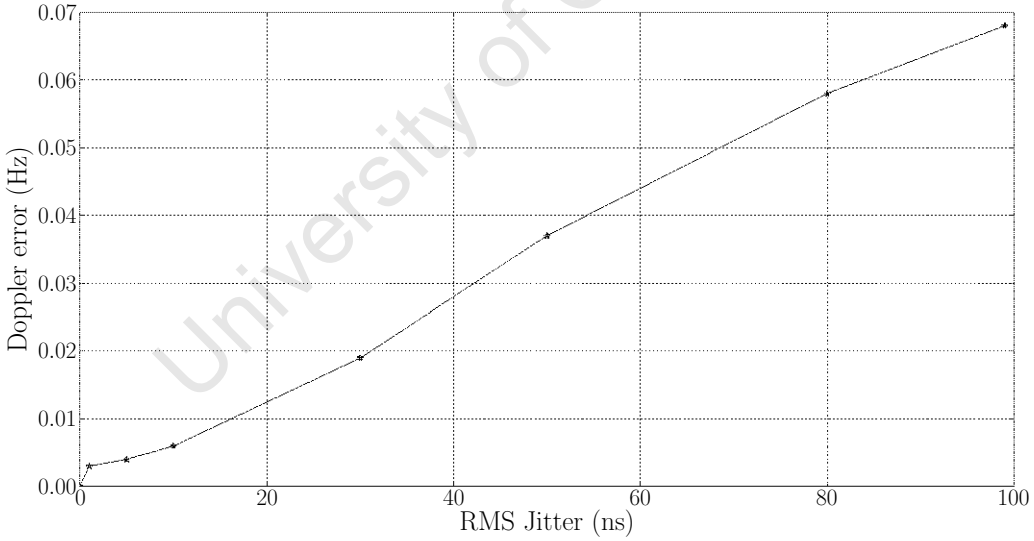
Rise time and offsets are also hardware induced properties of digital signals [28]. Rise time being the time taken for the digital pulse to switch from a 'low' state to a 'high' state (the opposite being fall time), and an offset being any fixed deviation of the pulse position from the intended output position.

A rise time of 20 ns is introduced onto the measurement gates, the range and Doppler measurements of 100 frames is shown in Figure 6.12. An offset of 10 ns is then introduced onto the early gate, results shown in Figure 6.13.

The results show that rise time and offsets on the measurements gates introduce a fixed offset in the range measurement and minimal distortion on the Doppler measurement. The range offset can be removed through calibration.

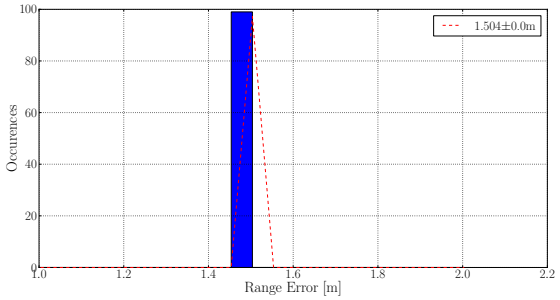


(a) Range

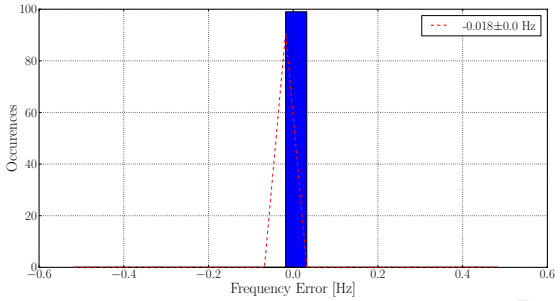


(b) Doppler

Figure 6.11: Python: Effect of jitter on radar measurements.

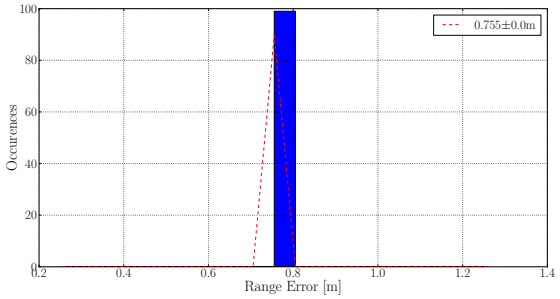


(a) Range

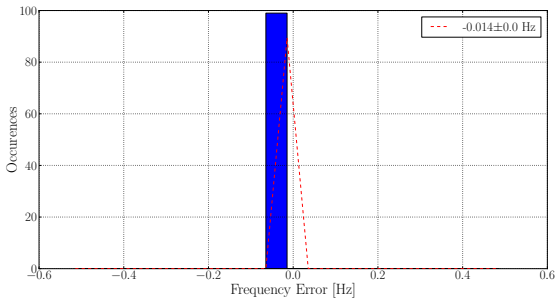


(b) Doppler

Figure 6.12: Range and Doppler measurements with 20 ns rise time on measurement gates.



(a) Range



(b) Doppler

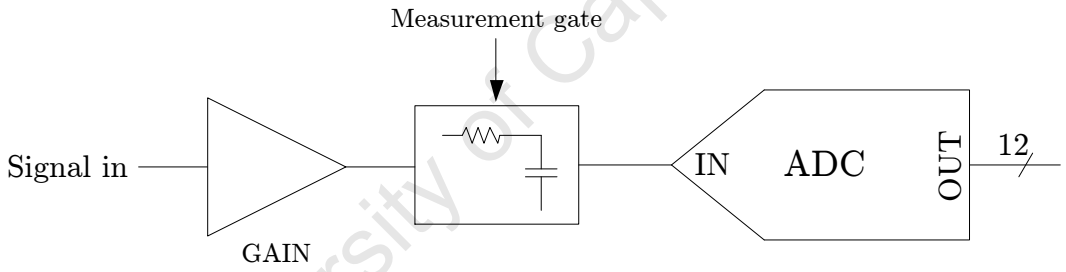
Figure 6.13: Range and Doppler measurements with 10 ns offset on measurement gates.

6.5 Proposed New SVCU Model

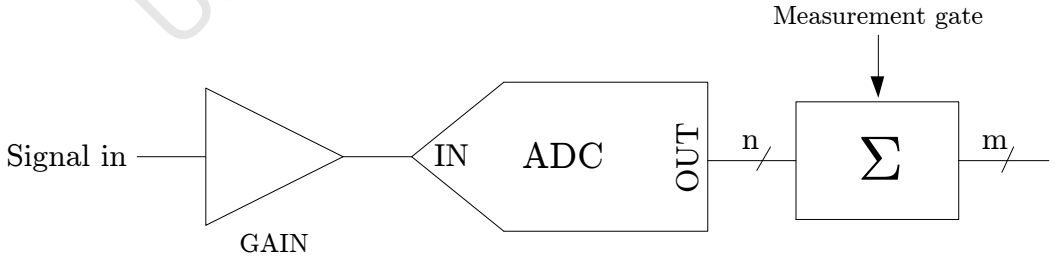
The modules/cards within the current SVCU were custom built for the project. In re-designing the SVCU, the strategy is to use Commercially available Off-The-Shelf (COTS) hardware wherever possible. In this section one such design is proposed. A timing study of the redesigned tracking channel is done, and the results measured against those of the old SVCU.

6.5.1 System Model

The current SVCU tracking channel consists of analogue integrate-and-hold circuits (driven by the measurement gates), followed by an ADC. In the new design the analogue signal will be digitized upfront by a high speed ADC, the resulting numbers will then be integrated digitally. The integration process will still be driven by the measurement gates. This is shown in Figure 6.14.



(a) Current configuration using analogue integrators.



(b) Proposed configuration utilizing digital integrators.

Figure 6.14: Configuration of the tracking channel within the SVCU.

ADC model

The ADC modelled for this study is the Texas Instruments (TI) ADS62P49 [13]. This is an eight channel, 14-bit, 250 Msps A/D converter. The ADC was modelled using the SystemVue^{TM}(SV) *AtoD* part configured to use a 'Jitter/INL/DNL' distortion model [12].

6.5.2 Non Linearity and Aperture Jitter Study

The SV tracking channel model was altered (new ADC and digital integration) and a calibration process was carried out to produce the new TR error models (as per Section 6.2).

Non linearity

Differential and integral non-linearities of 2 and 5 LSBs (from [13]) were then introduced into the ADC models. Figure 6.15 shows the effect on the TR range measurements. This fixed offset can be catered for through calibration.

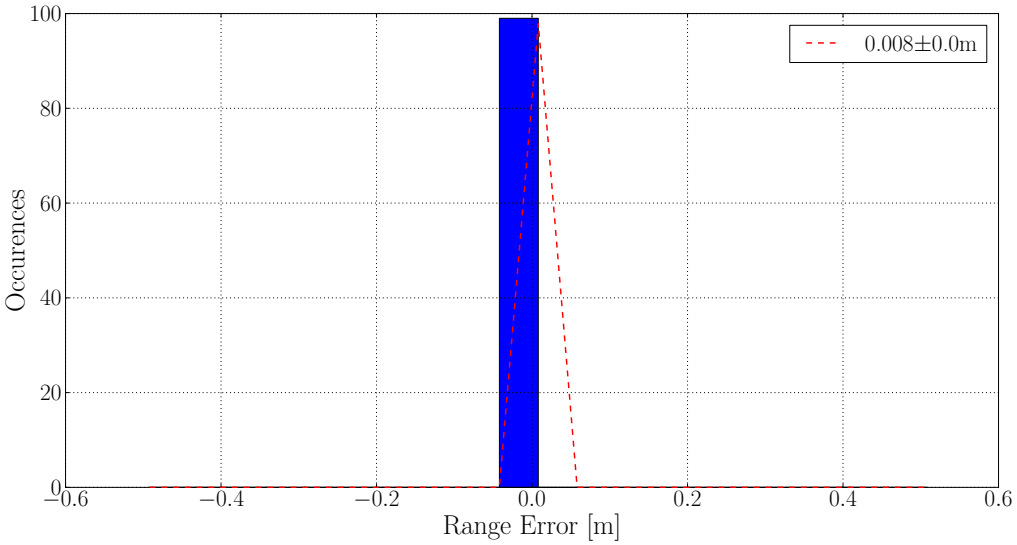


Figure 6.15: Python: Effect of ADC integral and differential non-linearity on TR range measurement.

Aperture jitter

Next aperture jitter (sampling uncertainty) was introduced into the ADC model by specifying it as a time domain RMS value (σ) in the SV *AtoD* part [12].

The TI ADS62P49 is specified with an aperture jitter of 145 fs (10^{-15}) or less [13]. The external sampling clock, used to drive the ADC, also introduces jitter into the sampling process. The total jitter is then the root-sum-square of the internal converter aperture jitter and the external sampling clock jitter [14].

The sampling clock used to drive the ADC will be a derivative of the TR master oscillator (MOSC, see Section 2.3.2 and 3.1.2). More specifically the 120 MHz MOSC will have to be approximately doubled to produce the 250 MHz sampling clock required by the TI ADS62P49.

Assuming an ideal frequency multiplier circuit, the phase noise of this new clock is given as [29]

$$\theta = \theta_{ref} + 20\log(N)\ \text{dB} \tag{6.5}$$

where θ_{ref} is the phase noise, in decibels, of the reference oscillator (in this case MOSC) and N the multiplication factor. Practically this means doubling the input clock results in a 6 dB increase in the phase noise of the output clock.

Figure 6.16 shows the single-sideband phase noise characteristic of the MOSC (formed from the TR documentation [7]) and the phase noise of a 250 MHz clock derived from the MOSC. By integrating the area under the graph of the derived clock, as per [14] and [15], the time domain jitter due to the sampling clock phase noise is calculated as 826 fs. Combining this with the ADC aperture jitter gives a total jitter of 839 fs for the ADC sampling process.

In the SystemVue$^{\text{TM}}$model, the aperture jitter was stepped up from 0 ps to 20 ps* producing Figure 6.17.

This shows that the 3 σ measurement errors due to aperture jitter are orders of magnitude smaller than the error due to the jitter on the measurement gates (section 6.4.1).

*SV limits the RMS aperture jitter value of the *AtoD* model such that $3\sigma \leq \frac{1}{16SSR}$ [12]. In the case of the TR simulation this limits σ to a maximum of 20 ps (given SSR = 1 GHz).

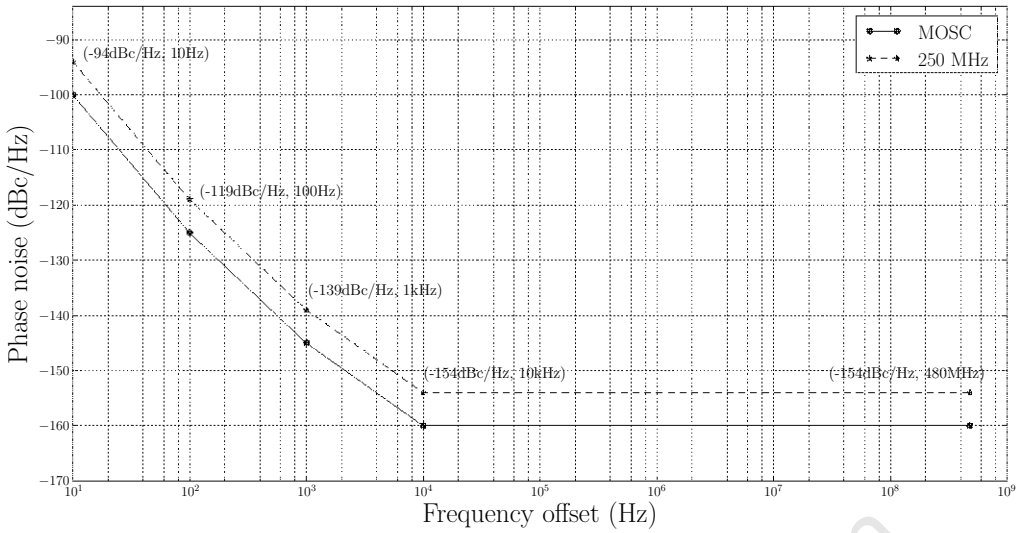


Figure 6.16: Python: Single-sideband phase noise characteristic of the MOSC and a clock twice the MOSC.

6.5.3 Digital Integration

The integration process within the modified SVCU model, though digital, are still driven or triggered by the three tracking gates. Jitter was introduced onto these gates, as in Section 6.4.1, to produce Figure 6.18.

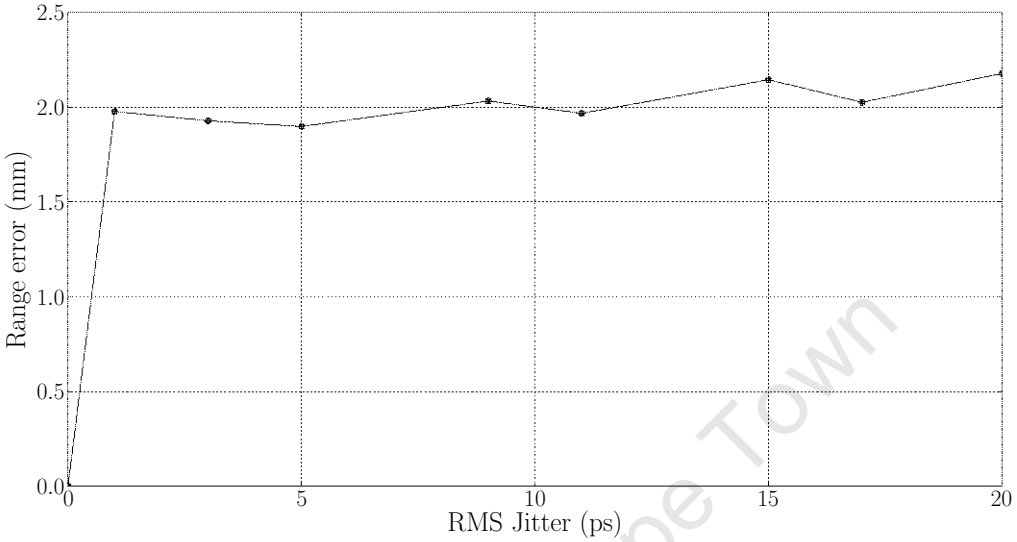
This shows that the jitter on the measurement gates will be as critical in the new design as it is in the current SVCU.

6.6 The Transmit Pulse

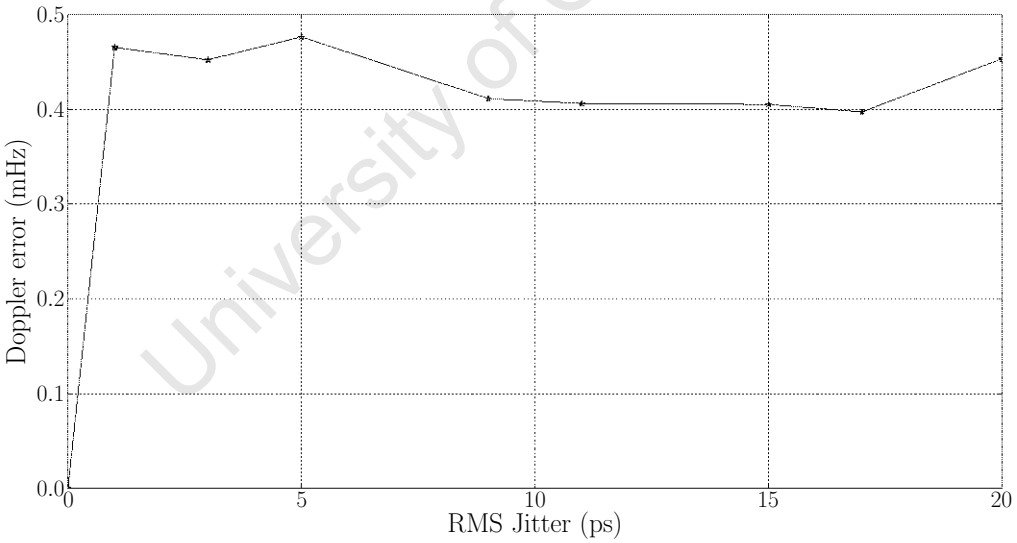
Thus far this study has largely ignored the TR transmit chain. The RF pulses transmitted by the TR are synchronized to the TX pulse from the SVCU (see Sections 2.3.1 and 3.2).

In order to simulate nanosecond jitter on the outgoing TX pulse, the whole SV TR model would have to be run at 1 GHz. This is beyond the memory resources of the computer used to build the model.

An assumption is made that if jitter were present on the TX pulse, this jitter will manifest in the target echo, propagate down the receive chain and present itself as jitter on the baseband video fed into the tracking integrators. Jitter, positional movement, on the baseband video is tantamount to jitter on the measurement gates.



(a) Range



(b) Doppler

Figure 6.17: Python: Effect of aperture jitter on radar measurements.

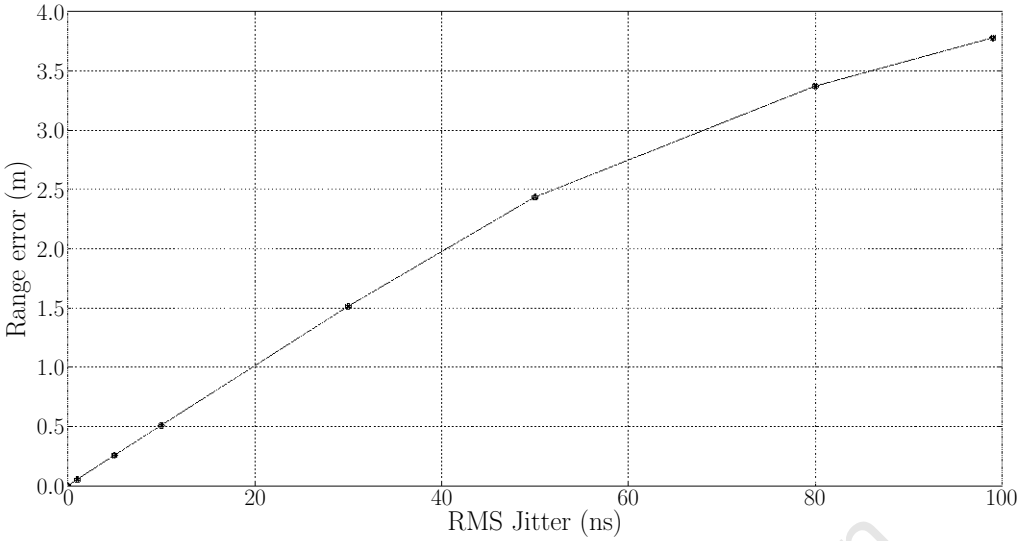


Figure 6.18: Python: Effect of jitter on digital integrators.

The TX pulse is thus also a timing critical synchronization pulse. The timely generation and transmission (to transmitter rack) of the TX pulse is critical to the accuracy of TR measurements.

6.7 The TR Jitter Budget

The TR is specified with a range measurement accuracy of 1.5 m *rms* (from Table 2.1). From the simulations it was found that the range accuracy deteriorates at approximately 0.5 m per 10 ns of jitter. The jitter budget (total jitter) on the measurement process within the TR is thus set at 30 ns.

The synchronization pulses that govern the measurement process are the TX pulse (synchronize transmitted RF pulses), the ADC sampling clock (aperture jitter) and the tracking/measurement gates. This translates to

$$\text{Total}_{jit} = \sqrt{{\text{TX}_{jit}}^2 + {\text{Aper}_{jit}}^2 + {\text{Gate}_{jit}}^2} \leq 30\text{ns} \tag{6.6}$$

In the next section a theoretical case study is done to determine if this jitter budget is achievable in an FPGA based SVCU design.

6.8 An FPGA Jitter Case Study

Modern FPGAs contain thousands of logic gates, are reconfigurable using Hardware Description Languages (HDL) and drastically reduce board space and costs (when compared to equivalent implementation using discreet logic) [30].

This case study investigates if the FPGAs available on the market today can meet the jitter requirements of the TR. The XILINX® Spartan-6 XC6SLX150T FPGA (see [31]) was chosen as the test FPGA for this investigation.

6.8.1 Overview of the Spartan-6 Architecture

All XILINX® FPGAs contain essentially the same basic logic resource, slices [31]. The Spartan-6 has 23038 slices, each comprising of four look up tables (LUT) and eight flip-flops. Slices are grouped (two) into common logic blocks (CLB) containing common arithmetic logic and registers. In addition to the logic resources the Spartan-6 also contains dedicated memory and clocking resources. Configuration of the clocking resources determine the timing, and hence jitter, characteristics of the FPGA and is expanded upon below [32].

Clocking resources

The Spartan-6 clocking resources consist of two clock networks and six† Clock Management Tiles (CMT).

Clock inputs It is recommended that all input clocks be input on one of the 32 (16 if differential) global clock pins (GCLK) on the Spartan-6. GCLK pins are normal I/O pins with special routing resources from the pin (periphery of FPGA die) to the CMTs (centre of die).

Manipulating clocks The input clocks are divided, multiplied or phase shifted by means of Clock Management Tiles. The Spartan-6 has six CMTs. Each CMT contains two Digital Clock Managers (DCM) and one Phase Lock Loop (PLL).

Clock distribution Clocks within the Spartan-6 can be distributed either on the I/O clock network or the global clock network.

The I/O clock network is a network of fast I/O clocks dedicated for I/O logical resources

†Numbers particular to Spartan-6 XC6SLX150T, for entire family see [31]

such as single or double data rate (SDR/DDR) interfaces.

The global clock network is a network of 16 high fanout clocks that is distributed to every synchronous element in the FPGA die. Unlike other families of XILINX® FPGAs, the Spartan-6 global clocks are not restricted to specific clock regions of the FPGA [32].

6.8.2 Generating the SVCU Timing Critical Signals in the Spartan-6

Here the jitter characteristics of generating the three sync pulses, governing the measurement process of the TR, utilizing the Spartan-6 clocking resources are presented.

ADC sample clock jitter

The TI ADS62P49 ADC requires a sample clock of 250 MHz. This clock must be generated within the FPGA using the 10 MHz TTL reference clock that is available to the SVCU (see Section 4.1).

To do this the reference clock will be input on one of the GCLK pins that route it to a CMT. Each CMT has two Digital Clock Managers (DCM). The DCM has a digital frequency synthesis output CLKFX, where [32]

$$f_{\text{CLKFX}} = f_{in}\frac{\text{M}}{\text{D}} \tag{6.7}$$

The CLKFX frequency is set by choosing an appropriate $2 \leq \text{M} \leq 32$ and $1 \leq \text{D} \leq 32$. In the case of this example $\frac{\text{M}}{\text{D}} = 25$.

The jitter on the DCM (DCM_CLKGEN primitive) CLKFX output is quoted as [33]

$$\text{CLKFX}_{jit} = \pm[0.2\% \text{ of CLKFX period} + 100] \text{ ps} \tag{6.8}$$

For a CLKFX output of 250 MHz, this works out to a jitter of 108 ps on the output clock. This clock can then be routed via an output pin to the ADC.

Measurement gate jitter

The measurement gates of the TR must be generated to a resolution of 6.25 ns [2]. To achieve this a 160 MHz clock will be required to clock the logic used to generate the gates. This clock is derived from the 10 MHz TTL clock in the same manner as the ADC sample clock. The resulting jitter on this 160 MHz clock, from Equation 6.8 (for $\frac{\mathrm{M}}{\mathrm{D}} = 16$), is 112.5 ps.

The 160 MHz clock is then routed via the global clock network to the logic that generates the measurement gate. In a synchronous design the jitter on the measurement gate will then be the root-sum-square of the clock jitter and IO jitter of the gate that outputs the measurement gate [34].

For a CMOS IO jitter of $\approx$ 35 ps (from [34]) this gives a total jitter on the measurement gates of 118 ps.

TX pulse jitter

The TX-to-TX time sets the PRI (and hence PRF) of the TR. The PRI is specified to the nearest microsecond and thus the TX is generated down to a resolution of 100 ns [21].

To do this, the 10 MHz TTL signal is fed into a DCM. The CLK0 output of the DCM is then used to clock the logic that produces the TX sync. The CLK0 output is a zero phase shifted output ($\sim$replica) of the input clock [32]. The jitter on the CLK0 output is given as [33]

$$\mathrm{CLK0}_{jit} = \pm 100 \text{ ps} \tag{6.9}$$

Combining this with the CMOS IO jitter gives a total jitter on the TX pulse as 106 ps (root-sum-square).

It must be noted that the jitter figure calculated above is at the output of the FPGA. The TX pulse is then sent from the SVCU to the transmitter rack via coaxial cable and a driver-receiver pair. The final jitter on the TX sync is thus the root-sum-square of the jitter at the output of the FPGA and the jitter introduced by this interface.

6.8.3 Jitter Budget of FPGA Based Design

The total jitter, in picoseconds, of the measurement process in a hypothetical FPGA based design is then

$$\begin{aligned} \text{FPGA\_T}_{jit} &= \sqrt{{\text{TX}_{jit}}^2 + {\text{Aper}_{jit}}^2 + {\text{Gate}_{jit}}^2} \\ &= \sqrt{(\text{Inter}_{jit}^2 + 106^2) + (108^2 + 0.145^2) + 118^2} \\ \text{FPGA\_T}_{jit} &= \sqrt{\text{Inter}_{jit}^2 + 36824}\ \text{ps} \end{aligned} \tag{6.10}$$

where Inter_{jit} is the jitter introduced by the interface routing the TX sync from the FPGA (SVCU) to the transmitter rack.

Equating this to the stipulated jitter budget of 30 ns reveals that Inter_{jit} must not exceed 29.99 ns for the TR to meet its range accuracy specification. This means in an FPGA based design the jitter introduced by the interface sending the TX sync to the transmitter rack will, almost entirely, determine the range accuracy of the TR.

6.9 Summary

A gated measurement (range and Doppler) process was implemented through the SystemVue$^{\text{TM}}$(SV) SVCU tracking channel model and Python signal processing scripts. The model yielded an accurate measurement of the target parameters to within 1 SI unit.

The tracking gates were then corrupted with jitter, rise time and offsets. Jitter on the measurement gates was found to be the most detrimental to the TR range accuracy, deteriorating at half a metre per 10 ns of jitter.

A new SVCU architecture was then proposed, using a fast ADC upfront and performing the tracking gate integration digitally. Once more jitter on the gates driving the digital integrators was the biggest contributor to measurement inaccuracy.

A case study to determine the jitter budget achievable in an FPGA based SVCU design was then done, concluding that: in an FPGA based design the jitter introduced by the interface sending the TX sync to the transmitter rack will, almost entirely, determine the range accuracy of the TR. The circuits used to transmit and receive the TX sync are thus critical. These circuits, and the components thereof, will have to be carefully selected,

tested and calibrated until the required performance (jitter, rise time and overshoot) is achieved.

Chapter 7

Conclusions and Future Work

This chapter discusses the conclusions drawn, based upon the results gained from the previous chapters. It is shown that the primary objectives have been achieved by building a working, system level, model of the SVCU. To validate this model the TR receive-transmit chain and signal processing were simulated. A critical timing study of the SVCU synchronization pulses was done and considerations for a SVCU redesign are addressed.

7.1 TR and Scene Model

A literature review of the TR and radar texts yielded an accurate system level model of the TR receive and transmit chain in SystemVue^{TM}. A target and scene model was then created and integrated into the TR transmit-receive model. The transmission, reflection (echo off target), reception and down-conversion to baseband of an RF pulse was then simulated.

7.2 SVCU Model

A circuit level SVCU model was developed and integrated into the receive chain model. The model encompassed three of the SVCU's critical functions: (1) digitizing the receiver video signals; (2) producing sync pulses for the radar sub-systems; (3) producing composite video for the A/R scopes.

7.3 TR Signal Processing

The TR signal processing (SP) routines were studied and an appropriate detector and gated tracking/measuring algorithm implemented in Python. The complete TR model, including SP, was run and validated through the accurate detection of a simulated target in range and Doppler.

7.4 Timing Study

A critical timing study was done by observing the measurement error when the measurement syncs (tracking gates) were corrupted. It was found that rise time and offsets produced constant errors, whilst jitter on the gates produced random measurement errors. The errors, range and Doppler, were measured as ± 0.5 m and ± 0.005 Hz per ten nanoseconds *rms* jitter respectively.

7.5 Considerations for SVCU Redesign

A new SVCU architecture was simulated, using a fast ADC (TI ADS62P49) upfront and performing the tracking gate integration digitally.

A timing study on this model found that:

- Differential and integral non-linearities within the ADC model produced fixed offsets in the range and Doppler measurements
- For an ADC aperture jitter of 20 ps a range error of ± 2 mm is predicted
- From the ADS62P49 data sheet and the TR master oscillator specifications, it was established that an aperture jitter of 839 fs could be achieved
- Jitter introduced onto the tracking gates that drive the digital integrators produced a range error of ± 0.5 m per ten nanoseconds

To meet the specified TR range measurement accuracy of 1.5 m *rms* (from [1]), the total *rms* jitter on the range measurement process must not exceed 30 ns.

The synchronization pulses that govern the measurement process are the TX pulse (synchronize transmitted RF pulses), the ADC sampling clock (aperture jitter) and the tracking/measurement gates. This translates to

$$\text{Total}_{jit} = \sqrt{{\text{TX}_{jit}}^2 + {\text{Aper}_{jit}}^2 + {\text{Gate}_{jit}}^2} \leq 30\text{ns} \tag{7.1}$$

7.5.1 An FPGA Based SVCU Design

A study was then done to determine the jitter budget achievable in an FPGA based SVCU design. The Xilinx Spartan-6 FPGA was chosen for this investigation. The study found that, through the careful use of the Spartan-6 dedicated clocking resources and a synchronous design, a total jitter of

$$\text{FPGA\_T}_{jit} \quad = \quad \sqrt{\text{Inter}_{jit}^2 + 36824}\text{ ps} \tag{7.2}$$

is achievable. Here Inter_{jit} is the jitter introduced by the interface routing the TX sync from the FPGA (SVCU) to the transmitter rack.

By equating this to the total jitter budget of 30 ns, we conclude that: in an FPGA based design, the jitter introduced by the interface sending the TX sync to the transmitter rack will, almost entirely, determine the range accuracy of the TR.

The circuits used to transmit and receive the TX sync are thus critical. These circuits, and the components thereof, will have to be carefully selected, tested and calibrated until the required performance (jitter, rise time and overshoot) is achieved.

7.6 Future Work and Recommendations

This section outlines the tasks to be done, going forward from this dissertation, to culminate in a newly built SVCU. Figure 7.1 shows a block diagram of a recommended architecture for a future SVCU, this forms the framework for the items discussed below.

1. Acquire a COTS board with reconfigurable logic (FPGA), memory and a microprocessor. The Reconfigurable Hardware Interface for computatioN and radiO

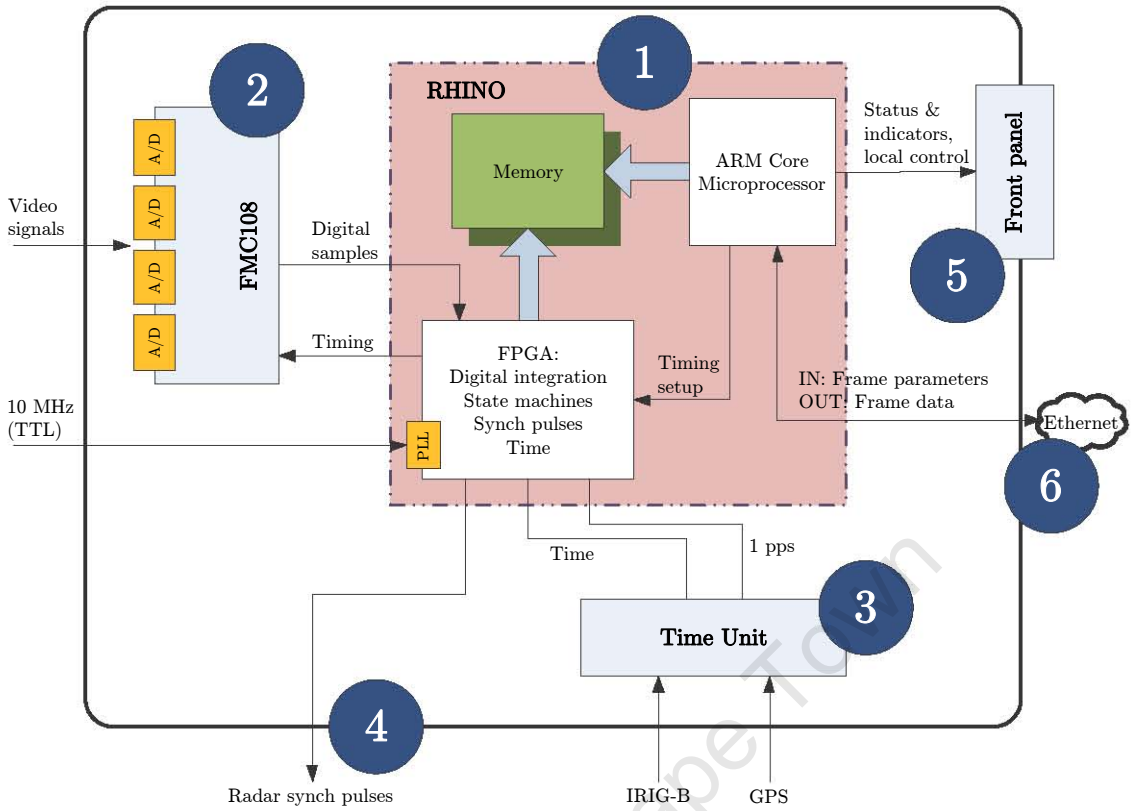


Figure 7.1: New SVCU design centred on the RHINO board.

(RHINO) board developed by the RRSG group at UCT is a good example [16]. The various state machines and data manipulation required by the SVCU can be implemented within the Xilinx Spartan-6 FPGA. A TI ARM Cortex-A8 processor also provides various peripheral functions such as USB, Ethernet and video out.

2. The analogue to digital conversion will be done by means of a FPGA Mezzanine Card (FMC) such as the FMC108 offered by 4DSP LLC [35]. The timing, control and transfer of data can be done utilizing the FPGA.

3. Time will be fed into the system using a commercial time server. Symmetricom (`http://www.symmetricom.com`) offer modules that use a GPS and-or time code synchronization source. These modules outputs various synchronized clocks and time in serial and parallel formats.

4. The synchronization pulses (syncs) output to the various TR subsystems will be generated within the FPGA. Driver and receiver circuits for these syncs will have to be designed. Special care must be taken for syncs driven over long lines and critical syncs such as the TX pulse.

5. A front panel must be built for status and fault indicators. It will also to allow local control of the SVCU.

6. The 1553 and HSD interfaces will be replaced by a single Ethernet link. Ethernet is a ubiquitous industry standard and will allow multiple systems to communicate with the SVCU.

Appendix A

Software Source Code

Listing A.1: SystemVue^{TM}MathLang script to form 20 range bins within the detection channel of the SVCU

```
outputReal = zeros(1, nRB * Pulses)
inpmat = zeros(Pulses, SamplesPRI)

index = 1

for j=1:Pulses
  inpmat(j,:) = input(index:index+SamplesPRI-1)
  index = index + SamplesPRI
end

index = 1

for j=1:Pulses
  detindex = detDelay
  for k=1:nRB
    outputReal(1,index) = sum(inpmat(j,detindex:detindex+PWSamples-1))
    index = index + 1
    detindex = detindex + RBSamples
  end
end
```

Listing A.2: SystemVue^{TM}MathLang script to simulate the signal processing within the Vector Processing Unit of the radar

```
FFTout = zeros(1, nRB*Pulses);
temp = zeros(nRB, Pulses);

R_Input = zeros(nRB, Pulses);
```

```
index=1;


for j=1:Pulses
  R_Input(1:nRB,j) = Input(index:index+nRB-1);
  index = index + nRB;
end

for i=1:nRB
   s_temp = R_Input(i,:);
   temp(i,:) = fft(s_temp,32);
end

index = 1;
for j=1:nRB
  FFTout(1, index:(index+Pulses-1)) = temp(j,1:Pulses);
  index = index + Pulses;
end
```

Listing A.3: Python script implementing **noncoherent detection** on data output from SystemVue^{TM}radar model (binary data file)

```
from __future__ import division
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import numpy as np
import matplotlib
import matplotlib.cm as cm
import matplotlib.font_manager as fm
from matplotlib.ticker import LinearLocator, FixedLocator,
    FormatStrFormatter
from pylab import rc

a = np.array(np.fromfile("NonCoherent.bin",dtype=complex))

matplotlib.rcParams.update({'font.size': 18, 'font.family' : 'serif'})

a = a.reshape(32,20)
ax1 = plt.subplot(111)
plt.plot((1/32)*(abs(a).sum(0)))
ax1.set_xlabel("Range bin")
ax1.set_ylabel("Sum of pulses")

plt.figure()
```

```
imgplot = plt.imshow(abs(a), origin = 'upper', cmap = cm.gray, extent
    =(0,20,0,32))

plt.figure()

ax1 = plt.subplot(111)

lfp = fm.FontProperties(size=10)

leg = []
for i in range(0,32,1):
    plt.plot(abs(a[i,:]))
    leg.append('Pulse: '+str(i+1))
ax1.set_xlabel("Range bin", size = 20)
ax1.set_ylabel("Range bin value", size = 20)
plt.legend(leg,loc='upper right',prop=lfp)
plt.legend

plt.figure()

ax1 = plt.subplot(111)
plt.plot(abs(a[0,:]))
ax1.set_xlabel("RangeBin")
plt.legend(['Pulse 1'],loc='upper right')

plt.show()
```

Listing A.4: Python script implementing **coherent detection** on data output from SystemVue™radar model (binary data file)

```
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.cm as cm
from matplotlib.ticker import LinearLocator, FixedLocator,
    FormatStrFormatter
import matplotlib
from pylab import rc


matplotlib.rcParams.update({'font.size': 14, 'font.family' : 'serif'})

a = np.fromfile("Coherent.bin",dtype=complex)

a = a.reshape(20,32)
```

```
upper = np.zeros((20,32))
lower = np.zeros((20,32))

thresh = 1000

upper = abs(a)>thresh
lower = abs(a)<=thresh
a[upper] = 1
a[lower] = 0

fig = plt.figure(2)
ax1 = plt.subplot(111)
imgplot = plt.imshow(abs(a), origin = 'lower', cmap = cm.gray, extent
    =(0,32,0,20))
ax1.set_xlabel("Doppler Bins",size=20)
ax1.set_ylabel("Range Bins",size=20)

fig = plt.figure(1)
ax = Axes3D(fig)

X = np.arange(0, 32, 1)
Y = np.arange(0, 20, 1)
X, Y = np.meshgrid(X, Y)

surf = ax.plot_surface(X, Y, abs(a), rstride=1, cstride=1, cmap=cm.jet,
     linewidth=0, antialiased=False)

ax.set_zlim3d(0, np.max(abs(a)))

ax.w_zaxis.set_major_locator(LinearLocator(10))
ax.w_zaxis.set_major_formatter(FormatStrFormatter('%.03f'))
ax.set_xlabel('Doppler Bins',size=20)
ax.set_ylabel('Range Bins',size=20)


fig.colorbar(surf, shrink=0.5, aspect=5)



plt.show()

# extract target data
x = a.argmax()
dims = a.shape
idx = np.unravel_index(x, dims)
```

```
print idx
```

Listing A.5: Python script written to analyse the effects of jitter on the radar measurements of range and Doppler

```
from __future__ import division
from csv import reader
from scipy import array, arange, fromfile, argwhere, std, mean
from scipy import sqrt, arctan, exp, pi, real, fft, zeros, imag
from scipy.interpolate import interp1d
from matplotlib import mlab, rc
import matplotlib.pyplot as plt

#setup matplotlib text
text = {'usetex' : 'True'}
font = {'family' : 'serif', 'serif' : 'computer modern roman', 'size' : '24
    '}
axes = {'grid' : 'True', 'labelsize' : '28'}
legend = {'fontsize' : '24'}
rc('text', **text)
rc('font', **font)
rc('axes', **axes)
rc('legend', **legend)

#-----------------------------------------------------------
#Functions
def nonull(stream):
   for line in stream:
        yield line.replace('\x00', '')
def readfile(filename):
    f = open(filename)
    ##f = open("EarlyToCalib.txt")
    read = reader(nonull(f))
    xaxis = []
    yaxis = []
    for row in read:
        try:
            xaxis.append(int(row[1]))
            yaxis.append(complex(int(row[2]),int(row[3])))
        except:
            continue
    return array(xaxis),array(yaxis)
#-----------------------------------------------------------

```

```
t = arange(-100,210,10) # add increment to end
rep = len(t)
f = arange(665,705,1)

rerr = (t-67.9585)*0.3/2
ferr = f-684

if(1):  #read files and reshape arrays to 21 x 32
    cgtstep,early = readfile("EarlyToCalib.txt")
    cgtstep,late = readfile("LateToCalib.txt")
    cgtstep,centre = readfile("CentreToCalib.txt")

    early = early.reshape(rep,32)
    late = late.reshape(rep,32)
    centre = centre.reshape(rep,32)

# Non coherent range
esum = (1/32)*abs(early).sum(1) #sum(1) collapses on row
lsum = (1/32)*abs(late).sum(1)

# Coherent range error
esum = []
lsum = []
for rrow in arange(0,rep,1):
    tempe = 0
    templ = 0
    for col in arange(0,32,1):
        tempe = tempe + early[rrow,col]*exp(-1j*2*pi*col*(684/2500))
        templ = templ +  late[rrow,col]*exp(-1j*2*pi*col*(684/2500))
    esum.append(tempe)
    lsum.append(templ)
esum = array(esum)
lsum = array(lsum)
err = real((esum-lsum)/(esum+lsum))

#Coherent range centre
csum = []
for rrow in arange(0,rep,1):
    tempc = 0
    for col in arange(0,32,1):
        tempc = tempc + centre[rrow,col]*exp(-1j*2*pi*col*(684/2500))
    csum.append(tempc)
csum = array(csum)

#Coherent Doppler error
```

```
edop = []
ldop = []
for i in f:
    tempedop = 0
    templdop = 0
    for col in arange(0,32,1):
        tempedop = tempedop + centre[0,col]*exp(-1j*2*pi*col*((i-5)/2500))
        templdop = templdop + centre[0,col]*exp(-1j*2*pi*col*((i+5)/2500))
    edop.append(tempedop)
    ldop.append(templdop)
edopa = array(edop)
ldopa = array(ldop)
doperr = real((edopa-ldopa)/(edopa+ldopa))

# create error models
interpra = interp1d(err,rerr,kind='cubic')
interpdo = interp1d(doperr,ferr,kind='cubic')
#-----------------------------------------

#Interpolation
cgt = fromfile("CGT.bin",dtype=complex)
egt = fromfile("EGT.bin",dtype=complex)
lgt = fromfile("LGT.bin",dtype=complex)

inidop = 684
realerrran = []
realerrdop = []

for r in arange(1,101,1):
    # reinitialize
    egat = 0
    lgat = 0
    edop = 0
    ldop = 0
    #Coherent sum (DFT)based on initial estimate of Doppler
    for col in arange((r-1)*32,r*32,1):
        egat = egat + egt[col]*exp(-1j*2*pi*(col-((r-1)*32))*(inidop/2500))
        lgat = lgat + lgt[col]*exp(-1j*2*pi*(col-((r-1)*32))*(inidop/2500))
        edop = edop + cgt[col]*exp(-1j*2*pi*(col-((r-1)*32))*((inidop-5)
            /2500))
        ldop = ldop + cgt[col]*exp(-1j*2*pi*(col-((r-1)*32))*((inidop+5)
            /2500))

    errorran = real((egat-lgat)/(egat+lgat))
    errordop = real((edop-ldop)/(edop+ldop))
```

```
    realerrran.append(interpra(errorran))
    realerrdop.append(interpdo(errordop))

realerrran = array(realerrran)
realerrdop = array(realerrdop)

ranerrmu =  mean(realerrran[1:1000])
ranerrsig = std(realerrran[1:1000])

doperrmu =  mean(realerrdop[1:1000])
doperrsig = std(realerrdop[1:1000])

#-------------------------------------------------------------
#Plots

#-#

#add figure
fig = plt.figure(facecolor='white')
#add
ax = fig.add_subplot(1,1,1)
#set lables
ax.set_xlabel('Range Error [m]')
ax.set_ylabel('Occurences')
#plot
n, bins, patches = plt.hist(realerrran[1:1000],20,normed=False)
y = mlab.normpdf( bins, ranerrmu, ranerrsig)
#add legends
##ax.legend.draw_frame(False)
#set boundaries
##ax.set_xbound(-20,20)
#add second axes
ax2 = plt.twinx()
#set lables
##ax2.set_ylabel('Range error distribution')
#plot
ax2.plot(bins, y, 'r--', linewidth=1.5)
ax2.get_yaxis().set_ticks([])
#add legends
ax2.legend([str(round(ranerrmu,3))+r'$\pm$'+str(round(3*ranerrsig,3))+'m'],
    loc='best')
##ax.legend.draw_frame(False)
#set boundaries
##ax2.set_ybound(ax.get_xbound)
```

```
#add figure
fig = plt.figure(facecolor='white')
#add
ax = fig.add_subplot(1,1,1)
#set lables
ax.set_xlabel('Frequency Error [Hz]')
ax.set_ylabel('Occurences')
#plot
n, bins, patches = plt.hist(realerrdop[1:1000],20,normed=False)
y = mlab.normpdf( bins, doperrmu, doperrsig)
#add legends
##ax.legend.draw_frame(False)
#set boundaries
##ax.set_xbound(-20,20)
#add second axes
ax2 = plt.twinx()
#set lables
##ax2.set_ylabel('Range error distribution')
#plot
ax2.plot(bins, y, 'r--', linewidth=1.5)
ax2.get_yaxis().set_ticks([])
#add legends
ax2.legend([str(round(doperrmu,3))+r'$\pm$'+str(round(3*doperrsig,3))+' Hz'
    ], loc='best')
##ax.legend.draw_frame(False)
#set boundaries
##ax2.set_ybound(ax.get_xbound)



plt.show()
```

Bibliography

[1] Lunette, *M2-1 General System Description Theory Book*. Denel Overberg Test Range, Arniston, South Africa, final ed., September 1992.

[2] Lunette, *M2-5 Synchronization and Video Conversion Unit (SVCU) Theory Book*. Denel Overberg Test Range, Arniston, South Africa, final ed., September 1992.

[3] M. I. Skolnik, *Introduction To Radar Systems*. McGraw and Hill, second ed., 1980.

[4] S. Kingsley and S. Quegan, *Understanding Radar Systems*. Scitech Publishing, 1999.

[5] M. A. Richards, J. A. Scheer, and W. A. Holm, *Principles of Modern Radar: Basic Principles*. Scitech Publishing, 2010.

[6] Agilent, "Systemvue electronic system-level design software." `www.home.agilent.com`, June 2011.

[7] Lunette, *M2-4 Receiving System Theory Book*. Denel Overberg Test Range, Arniston, South Africa, final ed., February 1994.

[8] Lunette, *M2-3 Transmitting System Theory Book*. Denel Overberg Test Range, Arniston, South Africa, final ed., October 1993.

[9] DOTR, "Computer Program Development Specification for SVCU Controller," tech. rep., Denel Overberg Test Range, 1984.

[10] Lunette, *M2-2 Signal Processing and Calibrations Theory Book*. Denel Overberg Test Range, Arniston, South Africa, final ed., October 1993.

[11] G. V. Morris, *Airborne Pulse Doppler Radar*. Artech House, 1988.

[12] A. Technologies, *SystemVue 2010.07 Manuals*. Agilent Technologies, `http://edocs.soco.agilent.com/display/doc/Home`, 2010.07 ed., 2010.

[13] T. Instruments, *Datasheet TI ADS62P49*. Texas Instruments, `http://www.ti.com`, slas635b ed., January 2011.

[14] W. Kester, *Converting Oscillator Phase Noise to Time Jitter*. Analog Devices, `http://www.analog.com`, a ed., 2009.

[15] J. C. LLC, "Phase noise (dbc/hz) to phase jitter (ps rms) calculator." `http://http://www.jittertime.com/resources/pncalc.shtml`, 2011.

[16] R. UCT, "Reconfigurable hardware interface for computing and radio." `http://www.ohwr.org/projects/rhino-hardware-01/wiki`, 2012.

[17] DOTR, "Product Specification Instrumentation Radar TR1," tech. rep., Denel Overberg Test Range, 1991.

[18] Lunette, *M2-16 Antenna System Theory Book*. Denel Overberg Test Range, Arniston, South Africa, final ed., September 1992.

[19] Lunette, *M2-10 Computer System Theory Book*. Denel Overberg Test Range, Arniston, South Africa, final ed., June 1993.

[20] DOTR, "SVCU Detail Design Specification," tech. rep., Denel Overberg Test Range, 1984.

[21] DOTR, *Requirement Specifications for MUXBUS I/O*. Denel Overberg Test Range, Arniston, South Africa, 2 ed., 2001.

[22] J. Collins and H. Huyzers, "SVCU Analysis," tech. rep., SimCon Global, 2011.

[23] E. Brookner, *Tracking and Kalman Filtering Made Easy*. John Wiley and Sons, 1998.

[24] E. W. Kang, *Radar System Analysis, Design, and Simulation*. Artech House, first ed., 2008.

[25] Lunette, *M2-8 Command and Control Assembly (CCA) Theory Book*. Denel Overberg Test Range, Arniston, South Africa, final ed., September 1992.

[26] D. Ascher and M. Lutz, *Learning Python*. O'Reilly, second ed., 2003.

[27] Open-source, "Python programming language - official website." `http://python.org/`, June 2011.

[28] T. Williams, *The Circuit Designer's Companion*. Newnes, second ed., 2010.

[29] W. F. Egan, *Practical RF System Design*. Wiley, 2003.

[30] R. Dubey, *Introduction to Embedded System Design Using Field Programmable Gate Arrays*. Springer, 2009.

[31] Xilinx, *Spartan-6 Family Overview*. Xilinx, 2.0 ed., October 2011.

[32] Xilinx, *Spartan-6 FPGA Clocking Resources User Guide*. Xilinx, 1.6 ed., May 2012.

[33] Xilinx, *Spartan-6 DC and Switching Characteristics*. Xilinx, 3.0 ed., October 2011.

[34] Xilinx, "Jitter of parallel output clocks in spartan-6." `http://forums.xilinx.com`, 2012.

[35] D. LLC, "Fmc108 high pin count fmc adc 8-channel 14-bit adc - 250 msps." `www.4dsp.com`, 2012.