A Radio Interferometer
Operating in the GPS L1 Band

by

Charles F. Shaw

A Thesis Submitted in
Partial Fulfilment of the
Requirements for the Degree of

Master of Science
in Electronics

at
The University of Otago
December 2013
To Neil, a beloved uncle,
who’s example to a small boy continues to inspire.

And to Thelma, my dear mother,
who encouraged me to reach for the stars.

In memoriam.
Abstract

The Transient Array Radio Telescope is a wide field synoptic synthesis array operating in the GPS L1 band. It is designed as platform for the development of new algorithms for radio astronomy. This thesis describes the development of this instrument.
Acknowledgements

This project would not have been possible without the support of many people. I am indebted to my supervisor Dr Tim Molteno for his guidance, support, patience, and enthusiasm. My am grateful to Phill Brown for his valuable advice and his assistance in helping me develop the technical skills necessary to complete this project. Special thanks go to my wife Stella, son Jonathan, and daughter Salani for their constant love and encouragement.
Contents

List of Figures ix

1 Introduction 1
   1.1 Background . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
   1.2 The Transient Array Radio Telescope . . . . . . . . . . . . . . . . 3
      1.2.1 Operating Band . . . . . . . . . . . . . . . . . . . . . . . . . 5
      1.2.2 Resolution . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
      1.2.3 Sensitivity and Signal to Noise . . . . . . . . . . . . . . . . 8
      1.2.4 Figure of Merit for Transient Detection . . . . . . . . . . . . 8
   1.3 Potential Applications . . . . . . . . . . . . . . . . . . . . . . . . 9
      1.3.1 Algorithm Development . . . . . . . . . . . . . . . . . . . . 10
      1.3.2 High Energy Cosmic Ray Detection . . . . . . . . . . . . . . 10
         1.3.2.1 Instruments . . . . . . . . . . . . . . . . . . . . . 12
   1.4 Outline of Thesis . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

2 Radio Front-End 17
   2.1 Antenna . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
      2.1.1 Antenna Noise Temperature . . . . . . . . . . . . . . . . . . . 20
   2.2 Radio Front End . . . . . . . . . . . . . . . . . . . . . . . . . . . 22
   2.3 Noise Performance of the Receiving System . . . . . . . . . . . . . . 29
      2.3.1 Noise Measurement Conventions . . . . . . . . . . . . . . . . . 29
         2.3.1.1 Noise Temperature . . . . . . . . . . . . . . . . . . 29
         2.3.1.2 Noise Factor and Noise Figure . . . . . . . . . . . . 29
         2.3.1.3 Converting Noise Figure to Noise Temperature . . . 30
      2.3.2 Calculating the Receivers Equivalent Noise . . . . . . . . . . . 31
         2.3.2.1 Comparison with Early Parametric Amplifiers . . . 33
      2.3.3 Overall System Noise . . . . . . . . . . . . . . . . . . . . . . 34
   2.4 Digital Sampling . . . . . . . . . . . . . . . . . . . . . . . . . . . 35
   2.5 Sensitivity . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 39
      2.5.1 The GPS Signal . . . . . . . . . . . . . . . . . . . . . . . . . 40
      2.5.2 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . 42
A.1.4  wfvm_avg.v ........................................... 116
  A.1.4.1  offset_meas.v ................................ 117
  A.1.4.2  data_sync.v .................................. 119
A.1.5  uart.v ................................................. 121
  A.1.5.1  mod_m_counter.v ............................... 122
  A.1.5.2  uart_rx.v ....................................... 122
  A.1.5.3  fifo.v ........................................... 125
  A.1.5.4  uart_tx.v ....................................... 127
A.1.6  control_buffer.v ..................................... 129
  A.1.6.1  dual_port_ram_sync.v ......................... 132
A.1.7  sig_to_rs232.ucf ................................... 133

B  Python Code for Base Station  135
  B.1  sig_acquire.py ..................................... 135
  B.1.1  observation.py .................................. 137
  B.1.1.1  telescope_config.json ....................... 139
# List of Figures

1.1 Dependence of Atmospheric Opacity on wavelength. ........................................ 2
1.2 Comparison of optical and radio images. ...................................................... 3
1.3 Two approaches for maximising resolution .................................................... 4
1.4 Components of the Transient Array Radio Telescope ..................................... 5
1.5 System block diagram ................................................................................. 6
1.6 Proton air shower above Chicago ................................................................. 11
1.7 Detection of Ultra High Energy Cosmic Rays ................................................. 13

2.1 Photo of the radio module: ........................................................................... 17
2.2 The GPS patch antenna and it’s radiation pattern ......................................... 18
2.3 Graphical representation of the convolution between the Intensity distribution and the antenna power pattern [31 fig 2.5 pg. 58] .......................... 20
2.4 The mounted antenna. .................................................................................... 21
2.5 Block diagram of the Radio Module .............................................................. 22
2.6 Chip inductor quality factor ........................................................................ 24
2.7 Radio front end schematic .......................................................................... 26
2.8 Remote station PCB layouts ....................................................................... 27
2.9 Remote station PCB photographs ................................................................ 28
2.10 Noise factors of components. .................................................................... 32
2.11 Quantisation noise .................................................................................... 36
2.12 ADC Transfer curve comparison. ................................................................ 38

3.1 Polar plot of fringe function. ....................................................................... 48
3.2 Block diagram of the clock distribution system ........................................... 49
3.3 Diode clipper ............................................................................................... 51
3.4 Schematic of the clock generator ................................................................ 52
3.5 Schematic of the Jitter Cleaner .................................................................... 54
3.6 The jitter cleaner bandwidth selection jumpers ............................................ 55
3.7 Jitter on the clock signal before and after conditioning ............................... 56
3.8 Oscilloscope trace of the clock signal before and after conditioning .......... 58
3.9 Phase difference between radio modules ...................................................... 59
3.10 Standard deviation of jitter before and after conditioning ........................... 59
3.11 Timing histograms before and after conditioning

4.1 Photo of the assembled base station
4.2 Block diagram of the base station systems
4.3 RJ-45 and CAT-6 pinout
4.4 Schematics of the base station and radio module transceivers
4.5 Power supply block diagram
4.6 Line Regulation for the Exar SPX3940
4.7 Power supply schematics
4.8 Photograph of the radio module’s power supply
4.9 Machining of prototype base station board
4.10 Base station board layout
4.11 Photographs of the base station board
4.12 Assembled base station PCB

5.1 Top level verilog blocks implemented in the FPGA
5.2 Verilog blocks of DLL
5.3 Timing diagrams for the DLL
5.4 State diagram of the control buffer finite state machine
5.5 Block diagram of UART module
5.6 The observation object

6.1 Spectrum of sampled data
6.2 Visibility of artificial source
6.3 Visibility functions recorded from astronomical sources
6.4 Proposed site for the TART, located on the Benmore Range
Chapter 1

Introduction

This thesis outlines the development of the hardware for a radio telescope using GPS receivers. This instrument has been conceived as a low cost platform for the development of new algorithms for radio astronomy. It pursues its design objectives by utilising high performance specialized components developed for consumer electronics.

1.1 Background

The radiation emitted by astronomical objects is not confined to optical wavelengths, but encompasses the entire electromagnetic spectrum. Radio telescopes allow many phenomena to be observed that are invisible to their optical counterparts, and are also less constrained by the Earth’s atmosphere [23].

Figure 1.1 relates the opacity of the atmosphere to the wavelength of the incoming radiation. It shows the atmosphere is transparent to radio at wavelengths between 5 cm and 10 m, permitting the operation of radio telescopes. The operating band of the instrument described in this thesis is marked at 19 cm.

The capabilities of a radio telescope are illustrated by Figure 1.2. At left, 1.2a shows an optical image centred on the galaxy 3C31, while 1.2b (right) shows the extra detail revealed by a picture taken by a radio telescope (the Very Large Array). The detail in Figure 1.2b is visible because of Hydrogen line emission at
Gamma rays, X-rays and ultraviolet light blocked by the upper atmosphere (best observed from space).

Visible light observable from Earth, with some atmospheric distortion.

Most of the infrared spectrum absorbed by atmospheric gases (best observed from space).

Radio waves observable from Earth.

GPS L1 Band (19cm) Used by TART

Figure 1.1: Dependence of Atmospheric Opacity on wavelength. Electromagnetic radiation at the radio end of the spectrum is relatively unaffected by the earth’s atmosphere. The TART operates in the GPS L1 band centered at 19 cm (Image courtesy of NASA)

Resolution While radio telescopes are less affected by atmospheric affects than their optical counterparts, their longer operating wavelength presents a significant hurdle. Instruments of very large aperture are capable only of very modest resolution. The largest single aperture radio telescope in the world is located at the Arecibo Observatory, Puerto Rico (Figure 1.3a), and has a diameter of 305 m.

Radio Interferometry overcomes the engineering limitations of constructing large aperture radio telescopes by combining the signals from a number of instruments to effectively form a single telescope. This allows the construction of a instrument of virtually unlimited effective aperture. The signals of telescopes located across and between continents may be combined using specialist VLBI techniques, and baselines of thousands of kilometres obtained!
Figure 1.2: Comparison of optical and radio images for the radio galaxy 3C31 (NGC 383). (a) is an optical image while (b) is a radio image at 20cm (1.4 GHz). Note the jets visible in (b) - astronomers believe these are fuelled by material accreting onto a super-massive black hole. (Images courtesy The STScI Digitized Sky Survey, and NRAO/AUI)

1.2 The Transient Array Radio Telescope

The Transient Array Radio Telescope (TART) is a wide field synoptic synthesis array that operates in the GPS L1 band. It is designed as platform for the development of new algorithms for radio astronomy. Figure 1.3 shows an overview of the TART. This thesis will be structured in an order that follows the signal from the antenna.
Figure 1.3: The photographs demonstrate two different approaches to maximising the resolution of a radio telescope. With a physical aperture of 305m, Aricebo (a) is the largest single aperture radio telescope in the world (Courtesy of the NAIC - Arecibo Observatory, a facility of the NSF). The VLA (b) is an example of an aperture synthesis telescope. Its resolution is not limited by the size of each individual parabolic antenna, but is defined by the area over which they are distributed (Courtesy of NRAO/AUI). The TART is also an aperture aperture synthesis telescope.
Figure 1.4: The Transient Array Radio Telescope shown with two of eight channels installed. Signals from active GPS patch antennas (1 and 3) are digitised by radio modules (2 and 4). The signal is sent to the base station via CAT-6 cables. The base station comprises the base station board (5), a Spartan 3 FPGA development board (6), a RS-232 to USB converter (7), and a Raspberry Pi computer (8). The Pi is connected to a network via Ethernet cable (9). Note that one of the radio modules (2) is shown with RF shielding removed for clarity.

1.2.1 Operating Band

A primary consideration when designing a radio telescope is the selection of an operating band. For this instrument, there are compelling reasons to select the GPS L1 band, centered on 1.5742 GHz.

In order to detect GPS signals, every GPS Radio Front End incorporates a very high performance Low Noise Amplifier (LNA). This allows optimisation of the figure of merit \( FOM_t \) and signal to noise ratio \( R_{sn} \) defined in Equations 1.3 and 1.2 respectively.

The GPS L1 navigation band is protected by legislation, and is almost silent. GPS signals themselves are very faint, as low as -166 dBw at ground level [4].

Large astronomical events such as supernovae generate a strong radio signal across the entire spectrum, including the L1 band.

The GPS control segment incorporates sixteen ground tracking stations to ac-
Figure 1.5: The TART consists of up to eight radio modules (4 shown) which are connected to a central base station via Cat-6 cables. Each radio module incorporates a clock conditioner, radio front end, and GPS patch antenna. The base station provides a clock signal (16.368 MHz) and power supply (24 VDC). The incoming data is buffered by the FPGA and transmitted over an RS-232 link to a PC.
accurately determine the orbital parameters of each satellite in the GPS constellation. This is necessary because a user’s position is calculated by determining the range of each visible satellite.

The ephemeris parameters broadcast by each satellite are accurate to 1 - 2 m[6]. This corresponds to angular uncertainty of about 20 arc-seconds, given an altitude of 20,000 km. This is very small compared to the resolution calculated in 1.2.2. This allows the GPS satellites to be used as guide stars for calibration purposes.

A even greater level of accuracy is provided by post-processed ephemerides, compiled for geodetic applications. These can be downloaded from the Crustal Dynamics Data Information System, and are accurate at the dm-level[6].

The embedded GPS data time and position stamps each observation precisely. The GPS time is transmitted every 6 seconds, at the beginning navigation data frame. The GPS time scale is maintained to be within one microsecond of UTC[4].

1.2.2 Resolution

The resolution that may be achieved by an interferometer is a function of wavelength and antenna separation. The resolution, $R$ is given by:

$$R \approx \frac{\lambda}{D}, \quad (1.1)$$

where $\lambda$ is the wavelength in metres, and $D$ is the baseline in metres[3, pg.81].

The proposed instrument operates at a wavelength of 19 cm with a maximum baseline of 100 m. Using Equation 1.1 the resolution is calculated as $0^\circ 6'33''$. For comparison the resolving limit of the unaided human eye is about $0^\circ 1'2''$. This calculation assumes $D = 3$ mm, $\lambda = 740$ nm, and $R \approx 1.22 \times \lambda/D$ for a circular aperture. The Sun has an angular size of about $32'$. 


1.2.3 Sensitivity and Signal to Noise

Factors that affect the Signal to Noise Ratio of a radio telescope also define its sensitivity. The signal to noise ratio, $R_{sn}$ is defined as;

$$R_{sn} = C \frac{T_a}{T_s} \sqrt{\Delta \nu \tau},$$  \hspace{1cm} (1.2)

where $C$ is a constant, $T_a$ is the antenna temperature, $T_s$ is the system temperature, $\Delta \nu$ is the bandwidth, and $\tau$ is the signal averaging time [31][pg.10].

Two factors contribute to the system temperature $T_s$ : The receiver temperature $T_r$ and the antenna temperature $T_a'$. $T_r$ is a measure of the the internal noise present in the receiver electronics, while $T_a'$ represents the unwanted noise received by the antenna from external sources (as opposed to $T_a$ - the signal strength of a source)

The antenna temperature $T_a$ is related to the flux density received from the source. Increasing aperture size and antenna efficiency both serve to increase $T_a$ and hence the signal to noise ratio.

If a signal is averaged for time $\tau$, it contains approximately $2\Delta \nu \tau$ independent samples, establishing the link between sample size and sensitivity.

The TART differs from most radio telescopes by having a wide field of view and averaging the signal over a brief period. This impacts sensitivity, but creates a niche for the TART in the observation of high energy, short duration radio events. The following section introduces a figure of merit for this type of instrument. A detailed assessment of the TART’s sensitivity is presented in Section 2.5.

1.2.4 Figure of Merit for Transient Detection

The mid term review of the Australian astronomy decadal Plan [12] highlights the need for interferometers to conduct transient surveys. The requirements of such an instrument are embodied in a Figure of Merit, $FoM_t$ [15]:

$$FoM_t = \Omega \left( \frac{A_e}{T_s} \right)^2 K(\eta W, t_p W),$$  \hspace{1cm} (1.3)
which is a function of the telescope sensitivity $A_e/T_s$, instantaneous solid angle $\Omega$, typical time duration of the transient $W$, event rate $\eta$, and the time per telescope pointing (dwell time) $t_d$. The function $K(\eta W, t_p W)$ incorporates the likelihood of detecting a particular kind of transient [15].

Unfortunately Hynman et al do not elaborate on $K(\eta W, t_p W)$. It is reasonable to assume that $\eta$ is an event rate per steradian. The probability of detecting a (very bright) transient therefore depends on the field of view, defined by the expression $\Omega \times K(\eta W, t_p W)$. However increasing $\Omega$ impacts sensitivity, because $\Omega$ and $A_e$ are interrelated by the expression;

$$A_e \Omega = \lambda^2,$$  \hspace{1cm} (1.4)

where $\lambda$ is the wavelength [16, pg.29]. Substituting this into Equation 1.3 yields;

$$F_oM_t = \frac{\lambda^4}{\Omega T_s^2} K(\eta W, t_p W).$$  \hspace{1cm} (1.5)

Writing the expression in this form highlights the challenges of designing a wide field synoptic radio telescope. Increasing the field of view by a factor of $k$ requires a reduction of $T_s$ by a factor of $\sqrt{k}$ to maintain $F_oM_t$. In practice this is difficult, so the TART trades sensitivity for increased field of view. This provides it with a figure of merit for transient burst searches [15].

### 1.3 Potential Applications

The TART is simply a hardware platform to assist the development of new algorithms, rather than a research instrument in its own right. However, the potential applications of these new algorithms are worth considering. One concerns the detection of Ultra High Energy Cosmic Rays (UHECRs), very rare subatomic particles with energies exceeding $10 \times 10^{18}$ eV.
1.3.1 Algorithm Development

The algorithms proposed for the Square Kilometre Array (SKA) are similar to those used since the 1970’s, and scale poorly. For a correlator with $N$ input signals there are $N(N - 1)/2$ cross-correlation products \[^{[37]}\]. Consider that the VLA with 27 antennas requires 351 Multiply-Accumulate (MAC) units to correlate each pair. Extending this to 100, 200, and 300 antennas would require 4950, 19900, and 44850 MACs respectively. The SKA proposes several thousand linked antennae, and relies on computing technology that has yet to be developed.

Processing the vast quantities of data produced by the SKA will require very high performance central supercomputers capable of 100 petaflops per second processing power. This is about 50 times more powerful than the most powerful supercomputer in 2010 and equivalent to the processing power of about one hundred million PCs. \[^{[27]}\]

An alternative approach involves the development of new algorithms that are better suited to the needs of the SKA. It is hoped that the TART will provide a platform on which these new algorithms can be tested.

1.3.2 High Energy Cosmic Ray Detection

The first evidence of UHECRs was detected by a particle detector array at Volcano Ranch in New Mexico in 1961. The energy calculated for this event far exceeds that of anything previously detected - about $10 \times 10^{20}$ eV. A second event was not detected until 1991, at Dugway proving ground in Utah. Researchers there used an air phosphorescence technique to detect an UHECR with an energy of $3.2 \times 10^{20}$ E. This corresponds to about 51 J, or the kinetic energy of a cricket ball travelling at 91 km·h$^{-1}$ \[^{[14]}\].

How close this is to the speed of light is illustrated by an example from Walker \[^{[35]}\]. He proposed that the UHECR and a photon leave at the same instant, and race each other over a distance of one light year. At the end of the journey, the UHECR would have fallen only 46 nm behind the proton!
Figure 1.6: An AIRES simulation of a cosmic ray shower formed when a proton with 1TeV (1e12 eV) of energy hits the atmosphere about 20km above the ground. The ground shown here is a 8km x 8km map of Chicago’s lakefront (Image courtesy COSMUS group at the University of Chicago)
Determining the origin of UHECRs is an active area of research. Current physical models are unable to explain their enormous energy. Cosmic rays produced by the sun are limited to energies of about $10 \times 10^{10}$ eV. Even supernovae shock waves can only account for energies up to $10 \times 10^{15}$ eV. Current proposals include exotic phenomena such as the interaction between dark matter particles and black holes [9].

What seems likely, however, is that they originate in our own galactic neighbourhood - within about 160 million light years. Particles travelling further than this distance are predicted to have energies below $50 \times 10^{18}$, a limit known as the Greisen-Zatsepin-Kuzmin (GZK) cut-off. It is believed that above this level, energy is lost through interaction with photons of the cosmic microwave background radiation. Current results tend to indicate the validity of the GZK cut-off [9].

The rarity of UHECRs detection has hindered both the development of a physical model of their formation, and definitive proof of the GZK limit. Cosmic rays exceeding $19 \times 10^{19}$ eV occur at a rate of about one per square kilometre per year. At an energy of $1 \times 10^{20}$ eV this falls to a frequency of about one per 100 years [36].

The 2010 Astronomy and Astrophysics Decadal Survey prioritizes the development of new instruments to further this research [22]. One such project is LUNSKA, an Australian interferometer for detection of lunar Cherenkov radiation [12].

1.3.2.1 Instruments

When a UHECR strikes the upper atmosphere, cascading showers of high energy particles are spread over a large geographic area. These particles interact with the earth’s magnetic field and atmosphere, before ultimately striking the ground. The UHECR is detected indirectly by observing the interactions of resulting particle shower. The common detection techniques may be divided into three main categories.

1. Detection of radio geo-synchrotron emission.
Figure 1.7: This diagram demonstrates a variety of methods for detecting an air shower initiated by an Ultra High Energy Cosmic Rays (and gamma rays) (Public domain image courtesy K.Berntohr).
2. Optical detection of atmospheric phosphorescence and Cherenkov radiation.

3. Ground based detection - Cherenkov radiation from water tanks etc.

The Pierre Auger observatory in Argentina is one of the world’s most significant facilities. It employs four banks of telescopes for detecting atmospheric phosphorescence and 3000 sq km of ground based water Cherenkov detectors [9].

The KArlsruhe Shower Core and Array DEtector (KASCADE-Grande) is another important facility, located in Germany. Attached to it is the LOFAR Prototype Station (LOPES) which aims to detect radio transients from air shower events using a digital radio interferometer. These are correlated against events detected by the KASKADEs water Cherenkov detectors. Electron-positron pairs generated in an air shower event emit synchrotron radiation as they are accelerated by the earth’s magnetic field [13]. The ANITA balloon borne radio interferometer is a similar project that detects radio-Cherenkov emission from the Antarctic ice [20].

1.4 Outline of Thesis

This thesis continues in Chapter 2 with a discussion of the radio front end design, tracing the signal from antenna to digitised output. The noise temperature of the receiving system is calculated, and used to determine the sensitivity of the TART. Conclusions are presented in Table 2.4 which provides the minimum sample size required to detect sources of diminishing strengths.

Chapter 3 describes the system for distributing a clock signal from the base station to the remotely located radio front ends. This signal is used to synchronise the sampling of the radio signal received by the antennas. The necessity of stable clock distribution is discussed, and measurements made of key performance criteria.

Chapter 4 describes the transmission of data and power over CAT-6 cable, and the rationale for this design decision. The power supply design is also explained with regard to criteria such as transmission losses, fault tolerance, and noise suppression.
The Verilog and Python software for buffering, transmission, and the remote storage of data is described in Chapter 5. The code is implemented on an FPGA, and a Raspberry Pi computer.

The thesis concludes with Chapter 6. This reflects on the progress to date, and the performance constraints of the TART’s current specification. A future direction for the project is discussed. This includes further hardware development, and installation at a remote location.
Chapter 2

Radio Front-End

Figure 2.1: Photo of the assembled radio module (RF shielding removed for clarity). The board is divided into three areas, each with their own ground plane. The centre third of the board is occupied by the power supply and the RS-422 transceivers. The clock conditioner and radio front end are located to the left and right of this respectively. The clock signal track runs along the bottom edge of the board. Status LEDs are provided for Power, Loss of Lock, Lock Detect, and Antenna Flag.

This chapter discusses the design of the radio front end, tracing the signal
from antenna to digitised output. The noise temperature of the receiving system is calculated, and used to determine the sensitivity of the TART. Conclusions are presented in Table 2.4 which provides the minimum sample size required to detect sources of diminishing strengths.

The radio front end is supported by a jitter cleaner, power supply, and data transceiver, as shown in Figure 2.5. The clock distribution system is discussed in detail in Chapter 3. Chapter 4 discusses the data and power distribution system.

2.1 Antenna

![Antenna Image](image)

**Figure 2.2:** (a) shows a photograph of the Taoglas A.01.C.301111 two stage 30dB active GPS antenna used for the TART. It has a diameter of 50 mm, and is screw mounted and center fed. (b) shows the radiation pattern of the antenna [29]. Section through the XY plane - $0^\circ$ at top. Note the nearly isotropic radiation pattern above the horizon - ideal for a synoptic wide field radio telescope.

An isotropic antenna allows the entire sky to be viewed at once, but this impacts sensitivity (see Sections 1.2.3 and 1.2.4). The response of a hypothetical isotropic antenna would be unaffected by the position of a source.
While an isotropic antenna is unrealisable, some designs come close - particularly those developed for GPS navigation systems. The operating band of the TART allows these antennas to be used. Designing an equivalent antenna from scratch is by no means a trivial exercise.

The visibility function recorded by the telescope contains information about the power pattern of the antenna as well as the source. Fortunately corrections can be made in the Fourier domain. Referring to Figure 2.3, the signal received by the telescope is:

\[
\int_{\text{source}} A(\theta' - \theta) I_l(\theta') d\theta'
\]  

(2.1)

The mirror image of the antenna pattern \(A(-\theta)\) is the Point Spread Function (PSF). By replacing \(A(\theta)\) with its mirror image \(A(\theta) = A(-\theta)\), the convolution integral (eq.(2.2)) is obtained:

\[
\int_{\text{source}} A(\theta - \theta') I_l(\theta') d\theta' = A(-\theta) * I_l(\theta) 
\]  

(2.2)

This is an important result, as it allows recovery of the intensity distribution \(I_l\) using the convolution theorem. The convolution theorem states:

\[
A(-\theta) * I_l(\theta) = \mathcal{F}\{A(-\theta)\} \times \mathcal{F}\{I_l(\theta)\} 
\]

\[
= \mathcal{F}\{A(-\theta)\} \times V(u) 
\]  

(2.3)

To reconstruct the intensity function, the convolution is undone by dividing the visibility function by the Fourier transform of PSF. The inverse Fourier Transform of \(V(u)\) is then taken to recover \(I(l)\). [31, pg.58].

The Taoglas antenna. A photograph of the chosen antenna is shown in Figure 2.2a, and its power reception pattern is in Figure 2.2b. This is an active antenna providing 30 dB gain, and incorporates a Surface Acoustic Wave (SAW) filter, and two cascading Low Noise Amplifier (LNA) stages. Out of band rejection is good, and the quoted noise figure is about 3 dB. [29]
Figure 2.3: Graphical representation of the convolution between the Intensity distribution and the antenna power pattern [31, fig 2.5 pg. 58]

Figure 2.4 shows an antenna fixed to its mount, and deployed on the roof of the University of Otago Science III building. The heavy concrete base provides stability in all wind conditions, yet easily allows the antennas to be placed in a variety of configurations. It is anticipated that future deployments will use a similar mount.

2.1.1 Antenna Noise Temperature

The radiation pattern of an ideal antenna would have no side or rear lobes. In this case the antenna temperature would reflect the strength of the source. Unfortunately the antenna temperature will also reflect the strength of terrestrial sources received through the rear and side lobes. This degrades the SNR of the system. The total temperature $T_a$ (K) of an antenna is:

$$T_a = \frac{1}{\Omega_a} \int_0^\pi \int_0^{2\pi} T_s(\theta, \phi) P_n(\theta, \phi) d\Omega,$$

(2.4)

where $\Omega_a$ is the beam solid angle, $T_s(\theta, \phi)$ is the brightness temperature of sources(s)
Figure 2.4: (a) shows a close up of the mounted GPS antenna with the cover removed. This is fixed to the end-cap of a piece of PVC pipe. The pipe is set in 20kg of concrete in a plastic former. The radio module electronics are housed within the pipe. The CAT-6 cable can be seen entering from the right. In the final implementation the cable will be protected by a system of irrigation pipes. (b) shows an array of these on the roof of physics building.

as a function of angle, $P_n(\theta, \phi)$ is the normalised antenna power pattern (dimensionless), and $d\Omega$ is an infinitesimal element of solid angle (sr) [16, pg.779].

This may be broken into contributions from a variety of sources. In this case, the antenna temperature may be divided into components from wanted and unwanted sources $T_a$ and $T'_a$ respectively (see Section 1.2.3).

Referring to Figure 2.2b, note that the gain of the Taoglas antenna is down by a factor of about $-10$ dB in the rear facing direction. Assuming that the earth is at a temperature of 290 K, the antenna temperature $T'_a$ may be calculated as;
\[ T'_A = \frac{1}{(2\pi)} (290) \times 10^{-(10)} \times (2\pi) \]

\[ = 29 \text{kHz} \quad (2.5) \]

This figure is reasonable, but nevertheless impacts the telescopes system temperature, a measure quantifying the receiving systems noise. This ultimately determines the SNR and sensitivity of the TART. A deeper discussion on the noise performance of the receiving system is presented in Section 2.3.

### 2.2 Radio Front End

![Diagram of Maxim radio front end](image)

**Figure 2.5:** The Maxim radio front end IC is provided with a conditioned clock signal and a 2.8V power supply. It down-converts, amplifies, and samples the incoming radio signal. The 2-bit sign/magnitude data is converted to RS-422 levels for transmission to the base station.

The electronics group at the University of Otago has developed systems for wildlife tracking using the Maxim MAX2769B. A desire to capitalize on this expertise has influenced it’s selection as the radio front end for the TART. Incorporated on the chip is the complete receiver chain, including a dual-input Low Noise Amplifier (LNA) and mixer, followed by the image-rejected filter, Programmable Gain Amplifier (PGA), Voltage Controlled Oscillator (VCO), fractional-N frequency synthesizer, crystal oscillator, and a multi-bit Analogue to Digital Converter (ADC)
The Maxim IC may be configured via a serial interface, or pre-set configurations selected by jumpers. Details of the configuration options selected are presented in Table 2.1.

A schematic of the radio front end and its ancillary components is presented in Figure 2.7. The placement and values of the resistors and capacitors used generally follows the manufacturers application circuit [17, pg.10]. These include the 0.1 µF and 0.1 nF supply voltage bypass capacitors (C17-C20, C22-C25), the PLL loop filter capacitors and resistor (C14, C15, R5), and the AC coupling capacitors (C12, C13, C16).

An active antenna connected is to the SMA connector X1. The inductor L2 is added to facilitate biasing. Its reactance $X_L$ is chosen to block the 1575 MHz L1 band input, yet allow the bias current provided by the ANTBIAS pin to pass freely. The AC coupling capacitor (C12) allows the signal to pass freely to the LNA input (LNA2, pin 25), but blocks the flow of the biasing current.

The Murata LQG15HS27NJ02 (L2) has a inductance of 27 nH, and is self resonant at 1.7 GHz. A graph of quality factor $Q$ vs frequency provided by the manufacturer is presented Figure 2.6. Resistance $R_L$ is assumed invariant with frequency, so the point of maximum $Q$ in the graph corresponds with maximum $X_L$, as shown by the following equation;

$$Q_L = \frac{X_L}{R_L} = \frac{\omega_0 L}{R_L},$$

where $\omega_0$ is the resonant frequency, and $L$ is the inductance.

The LNA output (LNAOUT, pin2) is routed externally to to allow the option of an external SAW filter, and in this instance passed via a capacitor (C13) to the mixer input (MIXIN, pin 5).

A 16.368 MHz clock signal that has been conditioned by the jitter cleaner is delivered to pin 15 via a coupling capacitor (C16). This is multiplied ($\times 96$) to 1571.328 MHz for low side Local Oscillator (LO) injection into the mixer. The 1575.42 MHz L1 band signal provided to MIXIN is down-converted to 4.092 MHz. It then passes through a 2.5 MHz bandpass filter before amplification by the PGA, and sampling at 16.368 MHz by the ADC.
Figure 2.6: Graph of the quality factor vs frequency of the LQG15HS series chip inductors. The device selected has a inductance of 27 nH, and is self resonant at 1.7 GHz - close to the 1.575 MHz GPS L1 frequency [18].

The sampled 2-bit sign/magnitude signal is finally passed from pins 21 and 22 to the RS-422 transceiver for transmission back to the base station (see Figure 4.4).

The power supply is divided into three sections which are isolated by ferrites L3 and L4. To ensure reliability and noise reduction, each of the VCC inputs are bypassed to ground by a capacitor placed as close as possible to the pin. The outputs ANTFLG and LD are connected to LEDs via (internally biased) transistors T1 and T2. These provide a visual indication that the antenna has been detected, and a lock to the clock signal obtained. As the circuits are identical, only LD is shown in the schematic.

The board layout is shown in Figures 2.1, 2.8 and 2.9. The centre third of the board is occupied by the power supply and the RS-422 transceivers. The clock conditioner and radio front end are located to the left and right of this respectively.

Isolating the sensitive radio front end from the noise introduced by the various switching components is a key consideration. To this end, a separate ground plane is provided for each section. These are connected at a single point by a 0 Ω resistor.
The top and bottom layer ground planes of the power supply are stitched together by vias. The jitter cleaner and radio front end are each provided with RF shields. Track widths have been maximised where possible to decrease capacitance. This is especially important for the clock line which runs along the bottom of the board. Sharp bends and transitions in track width have been avoided to minimise signal degradation resulting from reflections.

The Si5317 is a moderately fast digital circuit dissipating as much as 726 mW. Heat dissipation for the jitter cleaner is provided by thermal vias between its ground pad and the ground plane on the bottom layer. This ground pad layout is in the product data sheet [25, pg.42]. The operation of the jitter cleaner is discussed in Section 3.2. Similar vias are included for the ground pad of the MAX2769B. In this case they are specified by the manufacturer to provide an ultra-low-inductance connection to ground [17, pg.12]. As the power dissipation is only 90 mW, so they are not required for heat dissipation.

Table 2.1: The MAX2769B is set to one of its preconfigured states to avoid the complexity of programming through the serial interface. This table shows the details of state 2, set by connecting the PGM and SDATA pins to logic high and SCLK and \( \overline{CS} \) to logic zero [17, pg.17].

<table>
<thead>
<tr>
<th>Reference Frequency (MHz)</th>
<th>Reference Division Ratio</th>
<th>Main Division Ratio</th>
<th>I &amp; Q or I Only</th>
<th>Number of IQ Bits</th>
<th>I &amp; Q Logic Level</th>
<th>IF Center Frequency (MHz)</th>
<th>IF Center Bandwidth (MHz)</th>
<th>IF Filter Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>16.368</td>
<td>16</td>
<td>1536</td>
<td>I</td>
<td>2</td>
<td>CMOS</td>
<td>4.092</td>
<td>2.5</td>
<td>5th</td>
</tr>
</tbody>
</table>
Figure 2.7: Schematic of the radio front end.
Figure 2.8: The radio module is a two layer 100 × 50mm board. The top and bottom layers are shown in (a) and (b) respectively. Note the division of the board into sections, and the separation of ground planes. Top and bottom layer ground planes for the power supply are stitched together. All components are placed on the top layer of the board to facilitate the use of a reflow oven.
Figure 2.9: Photographs of the top (a) and bottom (b) layers of the remote station PCB.
2.3 Noise Performance of the Receiving System

The noise performance of the radio front end has a profound effect on the sensitivity of the telescope. It is typically described by a parameter termed the system temperature. This is comprised of the antenna temperature and the receiver temperature. The first describes the unwanted signal received by (and generated within) the antenna. The second represents the noise generated within the receiver electronics [31, pg.10]. In this section, conventions for characterising noise will be introduced and used to calculate the system temperature of the radio front end.

2.3.1 Noise Measurement Conventions

A variety of conventions are in use to characterise the noise of an electronic system. As mentioned above, noise temperature is more commonly used in radio astronomy. Noise Figure (NF) is another measure in general usage. The various conventions, and their conversions will be outlined. This allows comparison between instruments, and an evaluation of TARTs radio front end.

2.3.1.1 Noise Temperature

Noise may be expressed as a noise temperature provided the assumption is made that all noise is thermally generated. The familiar equation for the noise power of a resistor is rearranged to make temperature the subject.

\[ T_e = \frac{1}{kBG} \int_{0}^{\infty} N_a(\nu) \, d\nu, \]  

(2.7)

where \( T_e \) is the equivalent temperature of the Device Under Test (DUT), \( \int_{0}^{\infty} N_a(\nu) \, d\nu \) is the total noise power added by the device under test, \( k \) is the Boltzmann constant, \( B \) is the bandwidth, and \( G \) is the gain [2, pg.33]

2.3.1.2 Noise Factor and Noise Figure

The Noise Figure (NF) is a logarithmic measure of amplifier performance in general usage, commonly referred to in technical documentation. Noise calculations
generally require this to be expressed as a noise factor (F), as defined in Equation 2.9. It is important that these measures are not confused. The conversion between noise factor F and noise figure NF are made using the relationship;

\[
NF = 10 \log_{10} (NF) \leftrightarrow N = 10^{\frac{NF}{10}}
\] (2.8)

The noise figure may be understood using two equivalent definitions. North defined it as the ratio of the output noise power of a noisy amplifier to that of a perfect noise free amplifier of the same gain. Friis used the ratio of SNRs of the input to the output of an amplifier. Both Friis and North assume a matched resistor at a standard temperature (290 k) is connected to each amplifier input [34, p.99]. Using Friis’ definition, the noise factor F is;

\[
F = \frac{S_i}{N_i} / \frac{S_o}{N_o}
\] or \[
F = \frac{S_i}{N_i} \times \frac{N_o}{S_o}
\] (2.9)

where \( S_i \), \( S_o \), \( N_i \), and \( N_o \) are input and output signal and noise powers respectively. The output signal \( S_o \) may be defined in terms of the input signal and gain \( S_o = GS_i \). The output noise depends on the amplifier’s own internal noise, the input noise, and the gain \( N_a + GN_i \). The gain \( G \) is \( \frac{S_o}{S_i} \) is by definition. Rewriting gives

\[
F = \frac{1}{G} \times \frac{N_a + GN_i}{N_i}
\] (2.10)

\( N_i \) is typically expressed in terms of the noise power \( N_i = kT_0B \) that would be produced by a matched resistor at standard temperature (290 K) connected to the inputs. Substituting this expression for \( N_i \) gives;

\[
F = \frac{1}{G} \times \frac{N_a + kT_0BG}{kT_0BG}
\] (2.11)

This form of the equation has been adopted as the standard by the IEEE [21, pg.9].

2.3.1.3 Converting Noise Figure to Noise Temperature

A noise figure may be converted to a noise temperature using the relationship
\[ T_e = T_0 (F - 1), \text{ where } T_0 \text{ is } 290 \text{ K} \quad (2.12) \]

The noise temperature of a passive device may be calculated in a similar manner. Its noise factor \( F \) is equal to its attenuation \( L \);

\[ F = L, \quad (2.13) \]

assuming \( T = T_0 = 290 \text{ K} \) \cite{34, pg.536} where;

\[ L = \frac{1}{G_A}. \quad (2.14) \]

The equivalent noise temperature of an attenuator may therefore be expressed as;

\[ T_e = (L - 1)T_2, \quad (2.15) \]

\cite{34, pg.120} or alternatively using Equation \( 2.14 \)

\[ \left( \frac{1}{F} - 1 \right) T_2, \quad (2.16) \]

where \( T_2 \) is the physical temperature of the attenuator.

### 2.3.2 Calculating the Receivers Equivalent Noise

<table>
<thead>
<tr>
<th>Description</th>
<th>Gain (dB)</th>
<th>Gain</th>
<th>Noise Figure</th>
<th>Noise Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Antenna amplifier</td>
<td>30 dB</td>
<td>1000</td>
<td>2.9 dB</td>
<td>2.0</td>
</tr>
<tr>
<td>Co-axial cable</td>
<td>−3.4 dB</td>
<td>0.46</td>
<td>3.4 dB</td>
<td>2.19</td>
</tr>
<tr>
<td>Maxim LNA</td>
<td>13 dB</td>
<td>20</td>
<td>1.14 dB</td>
<td>1.3</td>
</tr>
<tr>
<td>Maxim Remainder Variable</td>
<td>Variable</td>
<td>Variable</td>
<td>10.3 dB</td>
<td>10.7</td>
</tr>
</tbody>
</table>

**Table 2.2:** It is necessary to convert the gain and the noise factor from logarithmic to ratio form in order to calculate the cascaded noise for the system.
Figure 2.10: Noise factors of components.
Friis’ noise equation allows calculation of the cascaded noise figure for the system, given $F$ and $G$ for each stage. If the logarithmic values $NF$ and $G_{dB}$ are provided by component manufacturers, they must be first be converted (See Equation 2.8 - $G_{dB}$ & $G$ similar). Friis’ equation is defined in [7, pg.31] as

$$F = F_1 + \frac{F_2 - 1}{G_1} + \frac{F_3 - 1}{G_1G_2} + \cdots + \frac{F_N - 1}{G_1G_2\ldots G_{N-1}}$$  \hspace{1cm} (2.17)

Substituting in the values from Table 2.2

$$F = (2.0) + \frac{(2.19) - 1}{(1000)} + \frac{(1.3) - 1}{(1000) \times (0.46)} + \frac{(10.7) - 1}{(1000) \times (20.0)}$$

$$= 2.0 + 0 + 0$$

$$= 2.0 \text{ (2.9 dB)}$$

This corresponds to an equivalent noise temperature of 290 K (Eq. 2.12).

It is clear the cascaded gain is dominated by the noise factor of the first amplification stage. The contribution of the noise factor of subsequent stages is reduced to zero by division by the 30 dB ($\times1000$) gain of the active antenna.

Lower noise active antenna alternatives were considered, but rejected because of their lower gain. Fine tuning of the noise performance could be facilitated by the use of a passive patch antenna with an external SAW filter. Equation 2.17 suggests that if the noise contribution from the SAW filter exceeded that of the LNA, it may be advantageous to place it after the first LNA stage. This is because the overall noise factor $F$ is dominated by $F_1$, the noise factor of the first component in a particular chain. If the gain $G_1$ is also high, the noise contribution from subsequent components will be negligible. The SAW filter is a passive device that does not provide gain.

### 2.3.2.1 Comparison with Early Parametric Amplifiers

In order to compare the radio front end with other instruments, it is advantageous to calculate the noise temperature of the radio front end without the active antenna installed;
\[ F = (1.3) + \frac{(10.7) - 1}{(20)} \]
\[ = 1.785 \quad (2.5 \text{ dB}) \]

This corresponds to a noise temperature of 228 K, comparing favourably with the early parametric amplifiers used for radio astronomy \[10\].

The development of parametric amplifiers in the early 1960s marked a critical milestone in the development of radio astronomy, and was instrumental in allowing observation at centimetre wavelengths \[23\]. De Jager in his 1962 paper discusses the development of such an amplifier at Dwingeloo for a telescope operating at 21cm. A noise temperature of 150 K is quoted for the front end (amplifier and mixer). This represented about a factor of five improvement over the pre-existing design \[10\].

### 2.3.3 Overall System Noise

Recall from Section 1.2.3 that the overall system temperature is comprised of the antenna noise temperature and the receiver’s equivalent noise. Friis’ equation gives an indication of the relative combination of each;

\[ F_{oa} = F_{ant} + \frac{F_{sys}}{G_{ant}} - 1, \quad (2.18) \]

where \( F_{oa} \) is the overall noise factor, \( F_{ant} \) is the antenna noise factor, \( F_{sys} \) is the system noise factor, and \( G_{ant} \) is the gain of the antenna. The noise factor \( F_{ant} \) of the antenna may be calculated by rearranging Equation 2.12, and substituting the antenna noise temperature \( T_a' \) calculated in Equation 2.5 as follows;

\[ F_{ant} = \frac{T_a'}{T_0} + 1 \]
\[ = \frac{(29)}{(290)} + 1 \]
\[ = 1.1 \]
The antenna gain $G$ of an antenna may be expressed in terms of its beam solid angle $\Omega$ (in steradians):

$$G = \frac{4\pi}{\Omega}, \quad (2.19)$$

[16], pg.26. The solid angle $\Omega$ (in steradians) subtended by a right circular cone is:

$$\Omega = 2\pi \left(1 - \cos \frac{\theta}{2}\right), \quad (2.20)$$

where $\theta$ is the apex angle.

$$G = \frac{2}{1 - \cos \left(\frac{90^\circ}{2}\right)} = 6.8$$

$$F_{oa} = 1.1 + \frac{(2) - 1}{4} = 1.35$$

This corresponds to an equivalent noise temperature of 102 K (Eq. 2.12).

Decreasing $\Omega$ increases the gain and reduces the overall system temperature, improving the signal to noise ratio and sensitivity of the telescope. Decreasing $\Omega$ also improves the SNR by increasing the strength of the signal (see Section 1.2.3).

As $G_{ant}$ is increased, the contribution to $F_{oa}$ of $F_{sys}$ is diminished, and the noise factor of the antenna $F_{ant}$ dominates.

### 2.4 Digital Sampling

In implementing a digital sampling system, consideration must be given to parameters such as sample size, quantization precision, and sampling rate. These impact telescope performance, bandwidth requirements, and memory utilization.
Figure 2.11: Sampling a signal with finite quantisation introduces quantisation noise which degrades the SNR of the system. The top panel shows the sampled signal superimposed over the input. The bottom panel shows the difference between the two. This is the quantisation noise.

Given that bandwidth and memory are finite resources, which parameter should be prioritised to maximise TART’s performance?

Key performance criterion for the TART are described Chapter 1. These are the signal to noise ratio $R_{sn}$, and figure of merit $FoM_t$, defined in Equations 1.2 and 1.3 respectively, and reproduced below;

$$R_{sn} = C \frac{T_a}{T_s} \sqrt{\Delta \nu \tau},$$

$$FoM_t = \Omega \left( \frac{A_e}{T_s} \right)^2 K(\eta W, t_d W).$$

The effect of sampling parameters on $R_{sn}$ is included in the constant $C$. The terms $A_e/T_s$ and $R_{sn}$ are broadly equivalent. Sample size is directly related to the signal averaging time $\tau$ and dwell time $t_d$. 

36
For purposes of illustration, suppose that the TART’s memory capacity can be doubled. This provides an opportunity to double the sample size, increasing $R_{sn}$ by a factor of $\sqrt{2}$ (41%). Assuming linearity, doubling $t_d$ also increases $K$ by a factor of two. Substituting $R_{sn} = \sqrt{2}$ doubles $(A_e/T_e)^2$, resulting in a total of $\text{FoM}_t$ by a factor of 4.

Alternatively, suppose the number of quantization levels $Q$ or the oversampling ratio $\beta$ could be doubled. Figure 2.11 illustrates the mechanism by which quantisation noise degrades the SNR of the system. To aid a performance comparison, Table 2.3 shows an efficiency factor $\eta_Q$ for various values of $Q$ and $\beta$. The efficiency factor relates the SNR of a signal with $Q$ quantization levels to that of a signal sampled with infinite precision.

<table>
<thead>
<tr>
<th>Quantization levels (Q)</th>
<th>$\eta_Q$ Sensitivity Relative to Unquantized Case</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\beta = 1$</td>
</tr>
<tr>
<td>2</td>
<td>0.637</td>
</tr>
<tr>
<td>3</td>
<td>0.810</td>
</tr>
<tr>
<td>4</td>
<td>0.881</td>
</tr>
</tbody>
</table>

Table 2.3: Efficiency factor $\eta_Q$ for various quantization schemes. $\beta$ represents the number of times oversampling, with $\beta = 1$ representing sampling at the Nyquist frequency. From Thompson et al [31, p.272]

Consider $Q = 2$ and $\beta = 1$ as a starting point. Doubling $Q$ yields a 38% increase in $R_{sn}$, and a 91% increase in $\text{FoM}_t$. Doubling $\beta$ yields a 17% increase in $R_{sn}$, and a 36% increase in $\text{FoM}_t$.

The results obtained above tend to indicate that quantization precision and oversampling rates should be traded off in order to achieve as large a sample size as possible.

Table 2.1 shows the configuration of the Maxim MAX2769B radio front end. A down-converted signal with an intermediate frequency center frequency of 4.092 MHz
and a bandwidth of 2.5 MHz is sampled with four quantization levels (2-bit) at 16.368 MHz. This corresponds to an oversampling ratio of about 1.5 times Nyquist ($\beta = 1.5$), and is not configurable. The quantization precision may be reduced however, by discarding the least significant bit.

While the TART has been designed to record 2-bit data, the LSB is presently discarded. This allows the length of recorded samples to me maximised. Two and four level (1 & 2-bit) ADC transfer curves are presented in Figure 2.12 for comparison. This shows that if the LSB of the 2-bit data (left) is ignored, a valid 1-bit transfer curve (right) is generated.

![Figure 2.12: Comparison of two level (1-bit) and four level (2-bit) ADC transfer curves. Note that discarding the LSB of the 2-bit data (left) generates a valid 1-bit transfer curve (right). $V_0$ is the threshold voltage.](image)

The Efficiency Factor Equations. Table 2.3 provides a comparison of the efficiency factors $\eta_Q$ of various sampling regimes. The values for the two-bit case are generated using the formulae described below. The reader is invited to refer to Chapter 8 of Thompson et al [31] for a detailed of their derivation, and for formulae for the two and three bit cases.

The efficiency factor $\eta$ for a two-level quantized correlation process relates its signal to noise ratio $R_{sn2}$, to that of an unquantized signal $R_\infty$ sampled at the same rate;
\[ \eta_2 = \frac{R_{sn2}}{R_{sn\infty}} = \frac{2\sqrt{\beta}}{\pi \sqrt{1 + 2 \sum_{q=1}^{\infty} R^2_2(q\tau_s)}} , \] (2.21)

where \( \beta \) is the oversampling rate (\( \beta = 1 \) at the Nyquist frequency), and \( R_2(q\tau) \) is a correlation coefficient defined as:

\[ R_2(q\tau_s) = \frac{2}{\pi} \sin^{-1} \left[ \frac{\beta \sin(\pi q/\beta)}{\pi q} \right] , \] (2.22)

where \( \tau_s \) is the sample interval, and \( q \) is an integer.

### 2.5 Sensitivity

The sensitivity of the TART may be expressed in terms of the weakest signal it can detect. The detection threshold is assumed to be the antenna temperature \( T_a \) for which signal to noise ratio \( R_{sn} \) is unity.

Inspecting Equation 1.2 indicates that \( R_{sn} \) is a function of sampling time \( \tau \), an easily adjustable parameter. Making \( \tau \) the subject allows calculation of the minimum sampling time required to detect a variety of sources;

\[ \tau = \left( \frac{R_{sn}}{C} \times \frac{T_s}{T_a} \right)^2 \times \frac{1}{\Delta \nu} . \] (2.23)

The antenna temperature \( T_a \) is defined as;

\[ T_a = \frac{S A_e}{k} , \] (2.24)

where \( S \) is the Power Flux Spectral Density (PFSD) in \( W/m^2/Hz \), and \( k \) is Boltzmann’s constant [16, pg.776].

The effective aperture \( A_e \) may be calculated from half power beam-width, and the beam solid angle \( \Omega \). The effective aperture \( A_e \) is;

\[ A_e = \frac{\lambda^2}{\Omega} , \] (2.25)
A visual inspection of the antenna radiation pattern in \[2.2b\] yields a half power beamwidth of \(\sim 90^\circ\). Using these two equations (\(\lambda = 19\) cm), the effective aperture is calculated as \(5.7 \times 10^{-3}\) m\(^2\).

The system temperature \(T_s\) and bandwidth \(\Delta\nu\) are 174 k and 2.5 MHz respectively. A value of 1 is assumed for the constant \(C\), correct within a factor of \(\sim 2\) \[31, pg.11\]. The PFSD \(S\) is known for a variety of astronomical sources. Table \[2.4\] presents the PFSD, \(T_a\), \(\tau\), and sample size computed for a variety of sources.

**Other factors influencing Sensitivity**

A stable clock ensures that the noise contributions from the Local Oscillator and A/D converter clock are insignificant compared to the thermal noise. For a detailed discussion of these factors, the reader is invited to refer to Thompson et al \[31, pg.233\], and an Agilent application note \[21\] respectively.

### 2.5.1 The GPS Signal

The GPS satellites orbit at an altitude of 20 183 km \[6, pg.164\], and broadcast a L1 band C/A signal with a centre frequency of 1575.42 MHz, and a bandwidth of 2 MHz\[6, pg.20\]. For the purposes of this discussion transmission power and the antenna gain are estimated as 25.6 W and 13 dBi respectively \[1\]. These values are consistent with the minimum received power levels published the GPS signal specification \[4\].

**Power Flux Spectral Density Calculation.** Multiple GPS L1 C/A signals will account for a significant portion of the power received by the TART. The Power Flux Spectral Density (PFSD) of these sources will be calculated and presented alongside that from astronomical sources.

Friis’ free space transmission formula defines the ratio of received power to transmitted power \(P_r/P_t\) as:
\[
\frac{P_r}{P_t} = G_t G_r \left( \frac{\lambda}{4\pi R} \right)^2, \tag{2.26}
\]

where \( \lambda \) is the wavelength (m), and \( R \) is the distance between receiver and transmitter (m), and \( G_t \) and \( G_r \) are the antenna gains of the transmitter and receiver respectively [2, pg.1758]. The gain \( G \) of an antenna is defined as;

\[
G = \frac{4\pi}{\lambda^2} A_e, \tag{2.27}
\]

where \( A_e \) is the effective aperture [16, pg.47]. Rearranging Equation 2.27 and substituting it into Equation 2.26 yields an expression for the received power \( P_r \);

\[
P_r = P_t \frac{G_t A_e}{4\pi R^2}. \tag{2.28}
\]

This may be rewritten by dividing by \( A_e \Delta \nu \) (where \( \Delta \nu \) is the bandwidth) to give an expression for Power Flux Spectral Density \( PFSD \), in \( W/m^2/Hz \) as;

\[
PFSD = \frac{P_t}{\Delta \nu} \times \frac{G_t}{4\pi R^2} \tag{2.29}
\]

Substituting the values above into equation 2.29 gives;

\[
PFSD = (25.6) \times \frac{(10^{13/10})}{(2 \times 10^6) \times 4\pi (20183 \times 10^3)^2}
\]

\[= 5.0 \times 10^{20} \text{ W/M}^2/\text{Hz}
\]

\[= 5.0 \text{ MJy} \tag{2.30}
\]

**Alternative PFSD Calculation.** The GPS interface specification quotes the terrestrial power levels of the Right Hand Circularly Polarised GPS L1 C/A signal are between \(-158.5 \text{ and } -150 \text{ dBW} \) at the output of a 3 dBi Linearly Polarised (LP) antenna [5]. These figures therefore include a 3 dB loss resulting from polarisation mismatch.
The received power is;

\[ P = 10^{\frac{P_{\text{dBW}}}{10}} \]
\[ = 10^{\left(-\frac{155.5}{10}\right)} \]
\[ = 281 \times 10^{-18} \text{ W} \]

The effective aperture \( A_e \) of a 3 dBi antenna \((G = 2)\) is;

\[ A_e = \frac{G\lambda^2}{4\pi} \]
\[ = \frac{(2)(19.02 \times 10^{-2})^2}{4\pi} \]
\[ = 5.76 \times 10^{-3} \text{ m}^2 \]

The power flux spectral density \( PFSD \) is;

\[ PFSD = \frac{P}{\Delta \nu A_e} \]
\[ = \frac{(281 \times 10^{-18} \text{ W})}{(2 \times 10^{-6} \text{ Hz})(5.76 \times 10^{-3} \text{ m}^2)} \]
\[ = 2.5 \times 10^{-20} \text{ W/m}^2/\text{Hz} \]
\[ = 2.5 \text{ MJy} \]

The upper bound of the GPS L1 signal PFSD corresponding to 150 dBW maximum received signal is 8.7 MJy. These figures support the result in Equation 2.30 which includes no allowance for atmospheric losses.

2.5.2 Conclusion

The TART is sufficiently sensitive to detect the GPS satellites, the Milky Way, the Sun, and the cosmic microwave background noise - even with a sampling duration of only 4 ms. Increasing the antenna gain, and the duration of each observation would improve the sensitivity, and allow the detection of fainter objects.
**Increased Observation Duration.** Radio telescopes such as the VLA are capable of making extended observations over the course of several hours. The Earth’s rotation must be compensated for in order to maintain a constant phase centre. This is accomplished by steering the parabolic antennae, and introducing a phase delay to the received signal (fringe stopping).

There is no means of adjusting the phase of the signal received by the TART’s fixed antennas. The Earth’s rotation tends to blur any image, limiting the resolution of the instrument over extended observations. Over the course of 5 min, a source traverses about 1°15′ of sky. This corresponds to the resolution limit imposed by an 8.7 m baseline.

Accepting this resolution penalty, it seems reasonable to (somewhat arbitrarily) set the upper limit on the duration of TART’s observations at 5 minutes. With no other changes other than increasing the buffer size, this would make Orion the faintest object detectable (see Table 2.4).

**Increasing Antenna Gain** Decreasing the the HPWB would also improve the sensitivity of the TART. Consider for example, a reduction from 120° to 60°.

\[
G = \frac{2}{1 - \cos\left(\frac{60°}{2}\right)} = 14.9 ,
\]

a factor of 3.7 improvement in the antenna gain. This is also true of the antenna temperature for any source that remains entirely within the antenna’s beam.

Increasing the antenna gain has the additional benefit reducing the system temperature. Substituting the revised gain figure into Equation 2.18 yields;

\[
F_{\alpha\alpha} = 1.1 + \frac{(2) - 1}{14.9} = 1.17 ,
\]

corresponding to a system temperature of 49 K, a factor of two improvement.
Further improvements could be made by decreasing the antenna noise temperature $T_a'$.

**Summary.** The sensitivity of the TART may be enhanced by increasing both the antenna gain and the observation duration. For the example above, simply reducing the HPBW from 120° to 60° would improve the sensitivity by a factor of nearly 8. Together an increase in the observation duration to five minutes, this would potentially allow the detection of the Andromeda galaxy. The effects of decreased sky coverage could be mitigated by using multiple antennas with overlapping fields of view.
<table>
<thead>
<tr>
<th>Source</th>
<th>PFSD</th>
<th>Ant Temp (k)</th>
<th>Samp Time (s)</th>
<th>Sample Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active Sun</td>
<td>38.7 MJy</td>
<td>323</td>
<td>$4 \times 10^{-8}$</td>
<td>-</td>
</tr>
<tr>
<td>GPS L1*</td>
<td>5 MJy</td>
<td>41.8</td>
<td>$2 \times 10^{-6}$</td>
<td>39 b</td>
</tr>
<tr>
<td>Gal. Centre</td>
<td>350 kJy</td>
<td>2.9</td>
<td>$5 \times 10^{-4}$</td>
<td>1 kB</td>
</tr>
<tr>
<td>CMB</td>
<td>-</td>
<td>2.73</td>
<td>$6 \times 10^{-4}$</td>
<td>1 kB</td>
</tr>
<tr>
<td>Quiet Sun</td>
<td>202 kJy</td>
<td>1.7</td>
<td>$1 \times 10^{-3}$</td>
<td>3 kB</td>
</tr>
<tr>
<td>Gal. Anti-centre</td>
<td>34.2 kJy</td>
<td>0.29</td>
<td>$5 \times 10^{-2}$</td>
<td>102 kB</td>
</tr>
<tr>
<td>Cas A</td>
<td>2.9 kJy</td>
<td>0.02</td>
<td>$7 \times 10^{0}$</td>
<td>14 MB</td>
</tr>
<tr>
<td>Cyg A</td>
<td>1.1 kJy</td>
<td>0.01</td>
<td>$5 \times 10^{+1}$</td>
<td>96 MB</td>
</tr>
<tr>
<td>Crab Nebula</td>
<td>692</td>
<td>0.01</td>
<td>$1 \times 10^{+2}$</td>
<td>243 MB</td>
</tr>
<tr>
<td>Moon</td>
<td>620</td>
<td>0.01</td>
<td>$2 \times 10^{+2}$</td>
<td>303 MB</td>
</tr>
<tr>
<td>Orion</td>
<td>479</td>
<td>0.00</td>
<td>$3 \times 10^{+2}$</td>
<td>508 MB</td>
</tr>
<tr>
<td>Vir A</td>
<td>132</td>
<td>0.00</td>
<td>$3 \times 10^{+3}$</td>
<td>7 GB</td>
</tr>
<tr>
<td>Andromeda</td>
<td>57</td>
<td>0.00</td>
<td>$2 \times 10^{+4}$</td>
<td>35 GB</td>
</tr>
<tr>
<td>3C 273</td>
<td>40</td>
<td>0.00</td>
<td>$4 \times 10^{+4}$</td>
<td>70 GB</td>
</tr>
</tbody>
</table>

**Table 2.4:** Table showing the Power Flux Spectral Density (PSFD) for various sources (in Janskys) [38, pg.240], and the minimum sample size required to resolve them above the system noise. The antenna temperatures are calculated assuming a half power beam width of 90° yielding an effective aperture of $11.5 \times 10^{-3} m^2$. The minimum sample time is calculated using Equation 1.2 assuming a system temperature of $102 k$ and a SNR of 1. Sample sizes are calculated for a sampling rate of 16.368 MHz. * See Section 2.5.1 for GPS signal strength calculations.
Chapter 3

Clock Distribution

This chapter describes the system for distributing a clock signal from the base station to the remotely located radio front ends. This signal is used to synchronise the sampling of the radio signal received by the antennas. The necessity of stable clock distribution is discussed, and measurements made of key performance criteria.

3.1 Background

A stable clock distribution system is an essential part of any digital radio interferometer such as the TART. The position of an astronomical source may be determined by very accurately measuring the phase difference of a signal received by a pair of antennas, located some distance apart. To make this measurement, the signal from each antenna must be sampled at precisely the same instant. These principles may be demonstrated by considering as an example, the following simplified interferometer.

Two hypothetical isotropic antennas are placed on an east-west baseline a distance $B$ apart. A single radio source with infinitesimal bandwidth traverses the sky from east to west directly overhead. Because the antennas are isotropic, the position of a source makes no difference to the received signal.

The signals received by the two antennas will drift in and out of phase as a
result of the changing difference in path length. This has the effect of casting an interference pattern over the sky.

This pattern is defined by the fringe function $F$;

$$F = \cos(2\pi\nu\tau_g) = \cos\left(\frac{2\pi B \sin \theta}{\lambda}\right),$$  

(3.1)

where $\theta$ represents the angle between the source and zenith, $B$ is the baseline in metres, $\lambda$ is the wavelength, $\nu$ is the frequency, and $\tau_g$ is the geometric time delay [31, pg.51]. A polar of this function in Figure 3.1 shows the alternating constructive and destructive fringes. The spacing is determined by $B/\lambda$, the length of the baseline in wavelength units. This ultimately determines the telescope’s resolution.

![Figure 3.1: Polar plot of fringe function. $B/\lambda$ is set to 3 for clarity. The + and - signs indicate positive and negative half cycles](image)

Introducing a phase delay to one of the antenna outputs has the effect of rotating the fringe function. This delay may be introduced deliberately in order to steer the beam, or inadvertently due to instabilities (clock jitter) in the clock distribution system. The latter impacts both the resolution and sensitivity of the interferometer.
Figure 3.2: A 16.368 MHz clock is supplied to an FPGA, and 8 radio modules (only one is shown). It is generated by a TCXO, and shifted to CMOS levels. The clock is converted to and from a differential signal for transmission over up to 100m of CAT-6 cable. A transceiver attenuates common mode noise. A jitter cleaner then reconstructs a clean clock signal and supplies it to the radio front end.

3.2 TART Clock Distribution System

An outline of the TART’s clock distribution system is presented in Figure 3.2. The master clock signal for the TART is generated by a Temperature Compensated Crystal Oscillator (TCXO) at the base station, and distributed to eight remote radio modules. It is also supplied to an FPGA, enabling sampling of the incoming data.

The design relies on the use of identical components and equal cable lengths to ensure equal phase delay and coherence between the clock signal provided to each of the radio front ends. A delay locked loop is implemented in the FPGA to
synchronise the data.

The RS-422 signal specification implements *differential signalling over balanced twisted pairs* (see §4.2), and provides a degree of immunity from external interference and cross-talk. Residual jitter is removed by a high performance digital phase locked loop located on the radio module. Schematics for the clock distribution system are presented in Figures 3.4 and 3.5.

**TCXO**

The TXC 7Q-16.368MBG fundamental mode TCXO selected for the TART is specifically designed to meet the needs of GPS applications. It produces a frequency output of 16.368 MHz that is stable to within ±0.5 ppm [33]. Unfortunately TCXOs with LVCMOS outputs are not available at this frequency. Instead this device produces a 0.8 V\textsubscript{pp} clipped sine wave output requires level shifting in order to drive LVCMOS logic.

A TCXO with a sine wave output is inconvenient for digital circuits because it has a much lower slew rate than a square wave. A small amount of noise on the signal will affect the timing of the low → high transition. This introduces timing jitter, negating the purpose of a accurate oscillator.

A common solution is to clip a TCXOs sine wave output close to the zero crossing point, where its slew rate is highest. This may be achieved using pair of clipping diodes as shown in Figure 3.3. Each diode is forward biased for one half of the cycle. A diode begins to conduct current when its transition voltage is reached, and the output voltage is limited to this level. The 0.8 V\textsubscript{pp} clipped sine wave output of the TXC device implies that diodes (perhaps germanium) with a transition voltage of 0.4 V have been used in the clipping circuit.

**Dual Inverter**

A dual inverter with Schmitt trigger inputs is utilised to shift the 0.8 V\textsubscript{pp} signal to a level compatible with CMOS logic. The use of a high speed analogue op-amp or comparator were also valid design options, and seriously considered.
Schmitt triggers are designed with hysteresis in order to provide a level of noise immunity. Because the rising and falling edges of the output are triggered at different thresholds, the duty cycle is slightly uneven. This small variation is tolerated by other components in the system, which trigger only on the rising edge of the clock.

The TCXO output is shifted to a mean level of 1.65 V (mid-rail) by a voltage divider network shown in Figure 3.4. A coupling capacitor isolates the TCXO from this DC voltage. A value of 10 pF is chosen to match the output load specified in the data sheet [33]. The input leakage current of the buffer is specified as 0.1 µA, so any voltage drop across the resistors in the voltage divider is negligible. Choosing 100 kΩ resistors limits the current flowing between VCC and GND. to 17 µA, minimising losses.

The 0.8 V_{pp} is marginal for driving the inverters. In order to address this issue, a 100 kΩ feed-back resistor is provided between the input and output. When a negative edge on the input drives the output high, the feedback resistor pulls the bias voltage up in preparation for the next positive edge on the input. The converse is true following a negative edge on the input.

**RS-422 Transceiver**

As the clock signal is transmitted over 100 m of CAT-6 cable it is subject to external interference and cross talk. The use of the RS-422 specification for balanced differential transmission allows removal of a large portion of this common mode interference.
Figure 3.4: Schematic of the clock generator. TCXO provides $0.8V_{pp}$ biased at mid rail by ac coupling capacitor and voltage divider. Inverter level shifts to LVCMOS voltages. Feedback resistor is necessary to ensure a fast output slew rate given the low input signal levels. CLK_OUT is passed to an RS-232 transceiver.

interference. Transceiver ICs provide the required conversion between LVCMOS to RS-422 signals at each end. Section 4.2 discusses the design rationale in greater detail.

**Jitter Cleaner**

While the RS-422 transceiver is effective in attenuating common mode noise, it is incapable of addressing random jitter. This results from thermal or external noise, and may be modelled as an independent random variable for each conductor. A Silicon Labs Si5317 jitter cleaning clock provides further attenuation of both random and deterministic jitter.

The Si5317 is a propriety digital PLL that incorporates a digital loop filter and Digitally Controlled Oscillator (DCO). The manufacturers of the Si5317 claim a spectacular RMS jitter output of 300fs [25]. It was chosen principally because it provides a high level of jitter attenuation without clock multiplication.
Configuration  The ease with which the Si5317 may be configured is an advantage. The operation frequency, loop bandwidth, reference clock type, and output signal are all set by tri-state jumpers. For the TART all of these except the loop bandwidth are permanently set. The schematic in Figure 3.5 provides details of the Si5317’s implementation in the TART.

A frequency range of 16.00 to 17.00 MHz has been chosen by setting FRQSEL = HHHM (see Figure 3.5). In this context H, M, and L represent inputs that are set high (3.3 V), mid-rail (1.65 V), and low (0 V) respectively. The loop bandwidth may be configured by setting BWSEL[1:0] jumpers in Figure 3.6 according to Table 3.1. A fundamental mode crystal is selected as the reference clock by setting RATE[1:0] to LL. Alternative settings allow the use of a third overtone crystal or an external clock.

The Si5317 has a flexible output driver structure that can drive a variety of loads, including LVPECL, LVDS, CML, and CMOS formats [25]. A CMOS output is selected by setting SFOUT[1:0] = LH. The TART uses only one of the Si5317’s two outputs. The second is disabled by a the application of a M to DBL2_BY.

It is possible to adjust the phase of the jitter cleaner output relative to its input. While this feature may be utilised in future designs, in the current implementation both INC and DEC are tied to ground.

Implementation  The Si5317 performs an internal self calibration following a Power On Reset (POR), or the application of a low to RST. The TART uses an RC network to hold RST low for 1 ms after power-up. This helps mitigate the effects of any transients on the power supply. A separate reset signal is considered unnecessary.

Power supply operation, and Loss Of Lock is indicated by the PWR and LOL LEDs shown in Figure 3.6. An output provided for a Loss Of Signal (LOS) alarm is left unconnected because this is also indicated by LLK. Correct operation of the interferometer is not possible in the LLK condition.

The incoming clock signal enters the jitter cleaner ckin+. The RS-232 transceiver output ro_clock is connected via a voltage divider network (R31 & R32), and cou-
and radio front end are each provided with RF shields. The jitter cleaner from the main power supply by a ferrite. Each of the VDD inputs are bypassed to ground by a capacitor placed as close as possible to the pin. The jitter cleaner and radio front end are each provided with RF shields.

The jitter cleaner is a digital circuit with a power consumption in the order of 200 mA. Care is required to ensure that it does not interfere with the operation of the radio front end, located on the same board. Its power supply is isolated from the main power supply by a ferrite. Each of the VDD inputs are bypassed to ground by a capacitor placed as close as possible to the pin. The jitter cleaner and radio front end are each provided with RF shields.

Figure 3.5: Schematic of the Jitter Cleaner.

pling capacitor (C5) as recommended in the product data sheet [25]. An Epson TSX-3225 38.4 MHz crystal is connected between XA and XB. It has a stability and accuracy of 10 ppm [26].
<table>
<thead>
<tr>
<th>Loop Bandwidth</th>
<th>BWSEL[1:0]</th>
<th>R20</th>
<th>R19</th>
<th>R18</th>
<th>R17</th>
</tr>
</thead>
<tbody>
<tr>
<td>6774 Hz</td>
<td>ML</td>
<td>15k</td>
<td>15k</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>1620 Hz</td>
<td>MM</td>
<td>15k</td>
<td>15k</td>
<td>15k</td>
<td>15k</td>
</tr>
<tr>
<td>400 Hz</td>
<td>MH</td>
<td>15k</td>
<td>15k</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>200 Hz</td>
<td>HL</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>99 Hz</td>
<td>HM</td>
<td>0</td>
<td>-</td>
<td>15k</td>
<td>15k</td>
</tr>
</tbody>
</table>

Table 3.1: The loop bandwidth is set by tri-state jumpers on the BWSEL inputs. This table shows the settings for frequency plan 79 centred on 16.5 MHz, taken from the Si5317 data sheet [25, pg.18].

Figure 3.6: BWSEL1 is set high or low by a single 0 Ω resistor at position R20 or R19. 15 kΩ resistors at both positions select medium. BWSEL0 is set similarly by resistors at R18 and R17. The LLK led warns the jitter cleaner is unable to phase lock to an incoming clock signal. The PWR led indicates operation of the power supply. R11 and R12 are not configurable.
Figure 3.7: Oscilloscope traces (a, c, & e) and (b, d, & f) show the clock signal before and after the jitter cleaner. Note the clear improvement in jitter, and also in slew rate. Each of these figures will be discussed in detail in the succeeding section.
3.3 Characterising Clock Performance

Ideally there should be no phase difference between the clock signal supplied to any pair of the TART’s radio front ends. In other words, the mean and standard deviation of the phase difference would be zero. Unfortunately phase error can not be eliminated, and this impacts the TART’s performance. The phase error is comprised of a fixed offset, a long term drift, and short term jitter. This section details how the clock errors are measured and how their effects on the telescope performance are quantified.

**Experimental setup.** To measure the phase error two radio modules were supplied a 16.368 MHz clock signal along two 20 m lengths of CAT-6 cable from a common base station. The phase of the two jitter cleaner inputs and outputs were compared first using an Agilent 54622D digital storage oscilloscope, and then with a Stanford Research Systems model SR-620 universal time interval counter. Interconnections were made with short equal-length coaxial cables, using SMA connectors provided at the jitter cleaner output. Otherwise, high quality oscilloscope probes were used.

One of the CAT-6 pairs was carrying 24 VDC originating from a switched mode power supply. Except as noted, measurements were made with no other data signals present in the pairs of the cable.

**Measurement using Oscilloscope.** One of the clock signals was used to trigger the oscilloscope, while the other was displayed in persistence mode. In persistence mode the display is not cleared as each new trace is written, allowing the short term phase variation to be observed. The timing jitter was indicated by the width of the trace. This was measured at 1.65 V using the oscilloscope’s cursors.

The jitter observed was the sum of the jitter on the displayed signal and the triggering signal. Using one of the clock signals as an external trigger ensured that the jitter observed was the quantity of interest, and not dependent on jitter in the oscilloscope’s internal time base. The jitter was measured at both the inputs and the outputs of the pair of jitter cleaners. This provided an indication of the
improvement made to the clock signal by the Si5317 jitter cleaner.

To measure the fixed phase offset of the pair of clock signals, both signals were displaced on the oscilloscope. Again one of these was used as the trigger, and the phase difference measured with the aid of the oscilloscope’s cursors.

The oscilloscope traces in Figures 3.8a and 3.8b show the clock signal at the input and output of the jitter cleaner respectively. The improvement made by the jitter cleaner is apparent. Figure 3.9a shows the fixed phase offset between a pair of jitter cleaned clock signals. This creates uncertainty in the position of the phase center of the telescope. This issue can be addressed in post calibration.

![Oscilloscope traces](image)

Figure 3.8: Oscilloscope traces (a) and (b) (1 ns/div) show the clock signal before and after the jitter cleaner. Note the clear improvement in jitter, and also in slew rate.

Measurement using Universal Time Interval Counter. The SR-620 was set to time interval mode, with a sample size of $5 \times 10^3$. The mean and standard deviation of the phase difference was read from the front panel display. Photographs are shown in Figures 3.10a, 3.10b and 3.9c.
Figure 3.9: A comparison of two conditioned clock signals shows a measurable phase difference. In each case the signal has traversed 20 m of CAT-6 cable. (a) shows the oscilloscope trace, while (b) and (c) show the phase difference in degrees and femtoseconds respectively.

Figure 3.10: Figures (a) and (b) show the jitter on the clock signal before and after the jitter cleaner. The measurement is made with a Stanford Research Systems Model SR 620 Universal Time Interval Counter. The relative jitter between clock signals is represented as a standard deviation in ns. The counter is locked to a reference provided by a Stanford Model FS725 Rubidium Frequency Standard.

In order to indicate the jitter distribution, a histogram was displayed using the SR-620’s XY oscilloscope outputs. A timing histogram of the jitter cleaner input and output is shown in Figures 3.11a and 3.11c respectively.

A second histogram was produced of the clock signal at the jitter cleaner input. In this instance sampled data from the radio front end was transmitted back to the base station along the same CAT-6 cable as used by the base station. This allowed the effect of cross-talk from other signal pairs to be observed on the jitter present on the clock signal. Figure 3.11 b & d compare timing histograms of the
Figure 3.11: The histograms (a) and (c) show the timing distribution of the clock signal before and after the jitter cleaner. Histograms (b) and (d) compare jitter on the clock signal without and with data transmitted alongside it. In each case, the jitter is measured at the input of the jitter cleaner. A comparison of the two indicates that crosstalk between signal pairs in the CAT-6 cable is not a particular problem.

jitter cleaner input with and without a signal present in one of the data pairs respectively.

3.3.1 Limits of Accuracy of the Measuring Instrumentation

The timing resolution $R$ of the SR-620 time interval counter is defined as;
\[ R = \pm \sqrt{\frac{(25 \text{ ps})^2 + (\tau_g \times \Delta f)^2 + (J_{\text{start}})^2 + (J_{\text{stop}})^2}{N}}, \quad (3.2) \]

where \( \tau_g \) is the gate time, \( \Delta f \) is the short term stability, \( J_{\text{start}} \) is the start trigger jitter, \( J_{\text{stop}} \) is stop trigger jitter, and \( N \) is the sample size [28, pg.6]. The numerator accounts for timebase and triggering jitter, and also includes an undefined 25 ps term that corresponds to the SR620’s single-shot resolution.

**Calculating the SR620’s Timebase Jitter** The internal TCXO is phase locked to the external Rubidium source by a PLL with a loop bandwidth of 20 Hz [28, pg.25]. Consequently, the timing performance of the SR-620 is defined by the internal TCXO for intervals less than 50 ms. This has an Allan variance of \( 2 \times 10^{-10} \) for intervals of one second [28, pg.viii]. The timebase jitter \( J_{\text{tbase}} \) is calculated for the measured delay of \( 57.253 \times 10^{-12} \text{ s} \):

\[
J_{\text{tbase}} = \tau_g \times \Delta f \\
= 57.253 \times 10^{-12} \times 2 \times 10^{-10} \\
= 11.5 \times 10^{-21} \text{ s}
\]

**Calculating the SR620’s Trigger Jitter** The trigger jitter is influenced by noise on the counter’s internal voltage reference, and the slew rate of the incoming signal [28]. The relationship is defined as

\[ J = \sqrt{\left( \frac{E_{\text{int}}}{(\frac{dV}{dt})_{\text{in}}} \right)^2 + \left( \frac{E_{\text{sig}}}{(\frac{dV}{dt})_{\text{in}}} \right)^2}, \quad (3.3) \]

where \( E_{\text{int}} \) is the internal noise (350 \( \mu \text{V}_{\text{rms}} \) typical), \( E_{\text{sig}} \) is the input signal noise, and \( (\frac{dV}{dt})_{\text{in}} \) is the input slew rate [28]. The slew rate of the jitter cleaner output is calculated as \( 891 \times 10^6 \text{ V/s} \) (from the published 20 – 80% rise-fall time of 2 ns [25]). An internal noise figure of 350 \( \mu \text{V}_{\text{rms}} \) is used as recommended [28]. Substituting these values into Equation (3.3):

\[
J = \frac{350 \times 10^{-6}}{891 \times 10^6} = 393 \times 10^{-15} \text{ s} \quad (3.4)
\]
Calculating the SR620’s Timing Resolution  Substituting the values for timebase and trigger jitter into Equation 3.2 allows the timing resolution \( R \) of the Stanford Research Systems SR-620 time interval counter to be calculated;

\[
\begin{align*}
R &= \pm \sqrt{\frac{(25 \times 10^{-12})^2 + (11.5 \times 10^{-21})^2 + (393 \times 10^{-15})^2 + (393 \times 10^{-15})^2}{5 \times 10^3}} \\
&= \pm 354 \times 10^{-15} \text{s}
\end{align*}
\]

(3.5)

3.3.2 Determining the Phase Centre Error

As discussed in Section 3.1, phase centre and resolution are determined by the relative phase delay and jitter between radio module reference clocks. These telescope parameters may be estimated using the values measured in Section 3.3.

Rearranging Equation 3.1 (using \( c = \lambda \nu \)) gives an expression for the phase centre error \( \theta \) depending on the timing error \( \tau \) in seconds;

\[
\theta = \sin^{-1} \left( \frac{c \tau}{D} \right),
\]

(3.6)

where \( c \) is the speed of light in metres per second, and \( D \) is the baseline in metres.

Phase Centre Offset  The phase centre offset is a constant angular displacement \( \theta_{\text{phcen}} \) of the phase centre from the zenith. Assuming a baseline of 40 m and a phase delay of 57.253 ns (see Figure 3.9b), the phase centre offset \( \theta_{\text{offset}} \) may be calculated as;

\[
\begin{align*}
\theta_{\text{offset}} &= \sin^{-1} \left( \frac{2.998 \times 10^8 \times (57.253 \times 10^{-12})}{40.00} \right) \\
&= 25.4^\circ \\
&= 25^\circ 24'0'' \pm 0.6''.
\end{align*}
\]

(3.7)

The error of \( \pm 0.6'' \) is calculated by setting \( \tau_g \) = 354 fs in Equation 3.6, using the result from Equation 3.5.
Phase Centre Uncertainty  Similarly, the phase centre standard deviation $\sigma_\theta$ may be calculated using the measured jitter standard deviation $\sigma_j = 29$ ps;

$$\sigma_\theta = \sin^{-1}\left(\frac{(2.998 \times 10^8) \times (57.253 \times 10^{-12})}{40.00}\right)$$

$$= 0.012^\circ$$

$$= 0^\circ 0' 45'' \pm 0.6''$$  

(3.8)

A comparison with the diffraction limited resolution puts this into context. For a baseline of 40 m using Equation 1.1, the resolution $R_{40m}$ is;

$$R_{40m} \approx \frac{19 \times 10^{-2}}{40}$$

$$\approx 0^\circ 0' 17''.$$  

(3.9)

3.3.3 Conclusion

Clock jitter is a significant factor in determining the resolution of the TART. The phase centre uncertainty $\sigma_\theta$ exceeds the diffraction limited resolution $R_{40m}$ by a factor of nearly three, as calculated from the measured jitter. This figure may be overly pessimistic, however. The measured jitter was about two orders of magnitude worse than the 300 fs claimed by Silicon Labs [25].

Jitter induced in the test leads may have been responsible for the discrepancy noted. While the use of short shielded cables minimised their contribution, jitter cannot be eliminated entirely. The use of active oscilloscope probes could address this issue, but were not available for this project.

Relying on the manufacturer’s data, the angular uncertainty may potentially be reduced to a best case of about one second of arc. An analysis of data from astronomical calibration sources may ultimately allow the angular uncertainty to be characterised more accurately.

The phase centre offset of 25$^\circ$ calculated in Equation 3.7 is indicative of what may be expected, and is not insignificant. The position of the GPS satellites are very accurately known, and they are strong sources in the L1 band. Using these
as calibration sources allows the offset to be compensated for during data post-processing. Preliminary data from the TART tends to indicate that any phase offset is stable over time.
Chapter 4

Data and Power Transmission

Figure 4.1: Photo of the assembled base station, responsible for transmission of data, clock and power to the radio modules. The power supply is at the left, and the connector for the FPGA development board is at the right. A total of eight RJ-45 connectors occupy the top and bottom edges of the board. The clock generator is located at the centre.

This chapter describes the transmission of data and power over CAT-6 cable, and the rationale for this design decision. The power supply design is also ex-
plained with regard to criteria such as transmission losses, fault tolerance, and noise suppression.

4.1 Background

The TART’s base station must record synchronous snapshots of data from eight spatially distributed radio receiver modules. In order to achieve this, consideration must be given to the following:

1. Transmission of clock and data signals over extended distances.
2. Distribution of a power supply.
3. A means of interfacing with a computer.

Clock and data signals are transmitted between each radio module and the base station at RS-422 levels, by three pairs of a CAT-6 cable. The remaining pair transmits 24 VDC. The base station itself comprises three main components (see Fig 4.2):

1. A base station board.
2. A Field Programmable Gate Array (FPGA).
3. A telescope control computer.

The first routes the data to the FPGA, and provides a clock signal and power to the radio modules. The second synchronously buffers the data, and transmits it over an RS-232 link. The third schedules the observations and saves the data to a remote network drive. This chapter focuses on the transmission of clock and data signals, and power distribution. Chapter 5 discusses the FPGA and control computer.
Figure 4.2: The base station distributes a 24V power supply and 16.368MHz sampling clock to eight GPS radio receiver modules. Incoming data is sampled and written to a remote network drive. This is facilitated by an FPGA and a telescope control computer.
In order to provide communication between the base station and radio modules, the TART requires a compatible system of cabling and line drivers. The implementation must meet the following criteria:

1. Able to transmit signals over the required distance (100 m).
2. Able to transmit signals at the required bit-rate (16.368 MHz).
3. Able to carry required signals in one cable (power, clock, and 2-bit data.)
4. Be economical and easy to work with.

### 4.2 Data and Clock Transmission

A system implementing RS-422 differential signalling over CAT-6 cable meets the requirements listed above. The cables are assembled in accordance with EIA/TIA-568A, as indicated in Figure 4.3. The non-standard pin and conductor assignments are shown in Table 4.1.

Various alternatives to CAT-6 cable were considered. Multi-mode fibre or coaxial cable would have both been workable solutions, and perhaps offered some advantages. However CAT-6 easily exceeds bandwidth requirements, is inexpensive, and simple to work with. It has the key advantage of being able to carry all of the required signals (including power) in one cable.

CAT-6 cable is designed to meet the requirements of Gigabit Ethernet where four Unshielded Twisted Pairs (UTP) each carry a 250 Mb/s data channel. The bandwidth requirements of the TART are somewhat more modest - one clock and two channels of data at 16.368 MHz. Unused pairs are often used for powering equipment such as routers, so powering the radio modules in this manner was straightforward.

CAT-6 cable achieves spectacular performance by utilising differential signalling over balanced twisted pairs. This technique has wide application in areas such as audio and telephony, and warrants further discussion.

In a balanced pair, neither conductor is connected to ground, and each conductor has an equal impedance to ground at any point along its length. This ensures
noise currents induced by a source in the far field are common to both conductors. Such common-mode currents are straightforward to cancel out.

While the use of parallel balanced pairs would be effective in eliminating noise from external sources, the issue of crosstalk remains to be addressed. Other signals carried within the cable are sources of noise in the near field. The small separation difference is significant in this case, and an unequal noise current should result. However, a common mode noise current is maintained by twisting each pair (at different rates). This ensures a constant average distance between signals and makes the balanced pairs resistant to both crosstalk and external noise.

When differential signalling is employed, the current carried by a pair of conductors is equal and opposite, and as a consequence so are the magnetic fields. Radiative losses are effectively eliminated, reducing attenuation over long cable runs. This prevents interference with other equipment, and crosstalk between signals - important considerations for a radio telescope.

Table 4.1: A CAT-6 EIA/TIA-568A ethernet cable carries power, clock, and data between the base station and each radio module. The (non-standard) assignment of pins and conductors is shown.

<table>
<thead>
<tr>
<th>Pin</th>
<th>Pair</th>
<th>Colour Code</th>
<th>Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>White &amp; Green</td>
<td>Data Magnitude Non-inverting</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>Green</td>
<td>Data Magnitude Inverting</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>White &amp; Orange</td>
<td>Data Sign Inverting</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>Blue</td>
<td>+24v</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>White &amp; Blue</td>
<td>Ground</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>Orange</td>
<td>Data Sign Non-inverting</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>White &amp; Brown</td>
<td>Clock Non-inverting</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>Brown</td>
<td>Clock Inverting</td>
</tr>
</tbody>
</table>
4.2.1 The RS-422 Transceivers

The clock and data signals are converted from single ended LVCMOS to differential RS-422 by a pair of Linear Technology LTC2851 RS-422 transceivers at either end of the cable. Each transceiver has one input and one output channel.

Each radio module receives a 16.368 MHz clock signal, and transmits 2-bit format sign/magnitude data. A second input to the radio module is unused. The differential inputs are tied to ground, and the corresponding output at the base station left unconnected.

A schematic is shown in Figure 4.4 of the RS-422 clock and data transceivers at the base station and radio module. The inverting and non-inverting driver outputs are pins Y and Z respectively. DI is the driver input, and RO is the receiver output. 100Ω termination resistors match the impedance of the CAT-6 cable, and are provided between the differential pairs to prevent reflections.

A key advantage of the LTC2851 transceivers is their ability to operate from a 3.3V supply. This is a departure from the norm - RS-422 levels are typically ±5 V. Switching noise is controlled by the addition of a 0.1uF capacitor, connected between VCC and ground. Two rows of transceivers can be seen in a photograph of the bottom side of the base station board (see Figure 4.11b).
Figure 4.4: Schematics of the base station and radio module transceivers, (a) and (b) respectively.
4.3 Power Distribution

The TART consists of (up to 8) radio modules connected to a base station by 100 m CAT-6 cables. Power supplies are derived from a 24 V DC system and distributed to each radio over an available twisted pair.

The power supply system must supply the required voltage and current, and be robust. Power supply noise must be minimised to ensure stable clock transmission, and fidelity of the sampled radio signal.

4.3.1 Extended DC Transmission

The effect of transmission losses in the 100 m DC supply lines is an important design consideration. Reliability, energy efficiency, and sizing of power supplies are all affected. Both the radio module power requirement and the cable DC resistance are not insignificant. These are 1 W and 6.7 Ω respectively.

Consider a potential scenario where a 3.3 V supply feeding the radio modules is located on the base station. Using the relationships $P = VI$ and $V = IR$, a voltage drop of 2 V is calculated. To avoid this situation, the 3.3 V supply is located on the radio module, and its input transmitted along the twisted pair at 24 V. This minimises current flow and limits the voltage drop in the cable to 0.3 V or 1%.

Obviously the 3.3 V and 2.8 V linear voltage regulators on each radio module can not operate directly from a 24 V supply. Excessive power loss and heat dissipation would lead to rapid failure! The 24 V is first stepped down to 5 V using a Murata Power Solutions OKI-78SR-5/1.5 switching regulator. This efficiency of this regulator ranges from 80 – 90%. The output ripple less than 34 mV

4.3.2 Fault Protection

The TART’s power supply system incorporates a number of layers of fault protection. These include protection against over-current, incorrect voltage and polarity, and voltage transients. Noise control is a particularly important element of the design.
Figure 4.5: A 24V supply is provided to the interface board radio modules (via CAT-6 cable). This is stepped down to 5V by a switching voltage regulator and then to 3.3V and 2.8V by a linear regulator. Protection is provided against polarity reversal, high voltage transients, and over-current.
**Fuses** Remote operation of the TART presents servicing challenges. In the event of a failure on one channel, data must be retrievable from the remaining good channels until a repair can be effected. This is facilitated by the protection of each channel with a dedicated fuse.

The power supply of the base station is also protected by a fuse. Unfortunately, an over-current fault here will bring the entire system down. A 500 mA slow blow surface mount fuse is utilised for the base station, and each of the remote stations.

**Incorrect Voltage and Polarity** The OKI-78SR-5 DC/DC converter used accepts an input from 7 to 36 volts, as this provides a certain level of robustness. They can also withstand a sustained short circuit without damage [19].

The DC/DC converter lacks reverse polarity protection. This is provided by the Schottky bridge rectifier (IC19) shown in Figure 4.7a. This device also allows safe connection of an AC supply.

**Transient Voltage Spikes** The base station provides two layers of protection against Transient Voltage Spikes as shown in Figure 4.7a. A bi-directional Transient Voltage Suppression (TVS) diode (D1) protects the bridge rectifier, and a uni-directional TVS diode (D4) arrests transients that may develop in the 24 Volt distribution system.

Similarly, a TVS diode (D4) protects the power supply of the remote station against transients in the incoming 24 volt line. This is shown in Figure 4.7b.

A measure of protection against transients is also provided by the capacitors C20 an C19 in Figure 4.7a and C29 and C30 in Figure 4.7b.

**Noise Control** Capacitors are provided between the power rail and ground at various points throughout the power supply to control noise. A common mode choke is provided at the point where the 24 V power supply enters the radio module (see Figure 4.7b). This helps to control EMI induced in the extended length of cable between the base station and the radio module.

The Murata Power Solutions OKI-78SR switching regulator has a voltage ripple on its output of about 34 mV [19]. An EXAR SPX3940 linear voltage regulator
makes the final reduction in voltage from 5 V to 3.3 V (REG2 or IC18 - see Figure 4.7). This type of regulator is chosen for its ability to reject voltage ripple on the input. This is specified as Power Supply Rejection PSR, and defined as:

\[ PSR = 20 \times \log \frac{R_i}{R_o}, \]  

(4.1)

where \( R_i \) and \( R_o \) are the input and output ripple voltages respectively.

Figure 4.6 reproduces the a graph of line regulation performance from the SPX3940 data sheet [11]. For an input of 5 V the slope of the line is about 1 mV/V. Using Equation 4.1, this equates to a (fairly typical) PSR of \(-60\) dB. The expected power supply ripple is now reduced to only 34 \( \mu V \).
Figure 4.7: Power supply schematics for the base station (a), and the radio module (b).

4.4 The Base Station Board

The base station board is a two layer 100 × 80 board. It can be easily fabricated in low volume by etching or milling. Initial prototypes were milled (Figure 4.9), while the final version was manufactured by an external board house (Figure 4.11).

The base station board was designed in Cadsoft’s Eagle PCB software. A free
Figure 4.8: Photo of the radio module’s power supply showing (1) Cable input - 24v (2) Common mode choke (3) Switching DC/DC regulator - 5v (4) Linear regulator - 3.3v. Note that this is a photograph of an early milled prototype board without the distinctive blue solder mask.

version allows board layouts up to 100 × 80, two signal layers, and is supported on multiple platforms. The selection of this software was influenced by a desire to make the TART easily accessible to the open source community.

The power supply is located at the right end of the board. Its two ground planes on the top and bottom of the board are separated from the board’s main ground plane for noise isolation. A single connection point is provided by a jumper.

The top layer of the board [4.11a] is dedicated to the routing of the 24 VDC supply and the clock signal. The clock generator is located at the centre of the board. An equal path length is provided for the clock signal between the clock generator at the centre of the board and each RS-422 transceiver. This is to minimise phase differences between the clock signal supplied to each radio module.

The bottom layer (Figure 4.11b) routes the clock and data signals between the
Figure 4.9: Machining of prototype base station board. All boards used for the TART are two layer to allow fabrication using easily accessible prototyping techniques such as etching or milling.

RJ-45 connectors and transceivers, and also data between the transceivers and FPGA connector at the left of the board.

All components were hand soldered to the board. Photographs of the finished board are shown in Figures 4.1 and 4.12.
Figure 4.10: Top (a) and bottom (b) layers of the base station board - dimensions $100 \times 80$mm.
Figure 4.11: Photo of the top (a) and bottom (b) sides of the base station PCB. The final board was manufactured by an external board house. Unlike the milled prototypes, these are supplied with solder mask, silk screen printing, through hole plating, and are tinned where appropriate.
Figure 4.12: Photographs of the top (a) and bottom (b) of the hand assembled base station PCB.
Chapter 5

Synchronous Buffering and Data Storage

The Verilog and Python software for buffering, transmission, and the remote storage of data is described in this chapter. The code is implemented on an FPGA, and a Raspberry Pi computer.

The Spartan-3 FPGA board is connected to the radio modules by a transceiver interface. It provides a 16.368 MHz clock and eight channels of two-bit sampled radio data, representing the sign and magnitude.

The FPGA samples and buffers the incoming signals before transmitting them over an RS-232 link to the Raspberry Pi. The magnitude bit is currently ignored, and less than eight channels are utilised. Figure 5.1 provides an overview of the functional blocks implemented in Verilog. Sections 5.1 to 5.3 will discuss their operation in detail.

5.1 Delay Locked Loop

Data must be sampled on the negative edge of a clock with matching frequency and phase. This ensures each bit is sampled only once, well away from its transition point. Various techniques may be implemented to provide a synchronous sampling clock. A clock signal may be transmitted with the data, or recovered from it using...
Figure 5.1: Top level verilog blocks implemented in the FPGA. The DCM multiplies a 50 MHz clock to 160 MHz, and feeds this to the DLL. The DLL uses this to synchronise the incoming data with an externally supplied 16.368 MHz clock. The data is sampled on the negative edge of this clock. The control buffer (activated by RX going high) buffers a snapshot of the incoming data and sends this in blocks to the UART. In the case shown, only four bits per sample are buffered. To avoid RS-232 framing errors, the remaining bits of the byte are set by an LFSR in rand_fill. The sampled data is transmitted down an RS-232 line (RX) to a computer.
Figure 5.2: The DLL aligns the data with the clock to ensure reliable sampling. The offset is first measured, and the data delayed by the appropriate amount. This is done to an accuracy of about 1/10 the period - adequate to ensure that the data is not sampled on the rising edge.

a Delay Locked Loop (DLL).

The TART maximises the data throughput of CAT-6 cable by implementing a DLL for clock recovery. The four conductor pairs available are utilised for power, clock, data sign, and data magnitude. Provision of a return clock signal would have allowed only one channel of data, reducing the sensitivity of the instrument.

The DLL is implemented in the FPGA, and comprises two functional blocks (see Figure 5.2). The first offset_meas measures the phase delay of the input data relative to the clock. The second data_sync applies a phase delay to ensure the output data is aligned with the clock. Timing diagrams are shown in Figures 5.3b and 5.3c respectively.

5.1.1 Removing Noise Transients from the Data Inputs

Noise transients on the input lines can cause data corruption. These are filtered by the wvfm_avg module which is implemented within both offset_meas and data_sync. The current output state is held until the input has remained stable.
Figure 5.3: The DLL adjusts the phase of the incoming data to match the sampling clock. This is done with sufficient accuracy to ensure the data is sampled well away from its transition. Sampling is on the negative edge of the sampling clock. (a) shows a timing diagram for the DLL. The DLL incorporates offset_meas and data_sync modules. (b) shows a timing diagram for the offset_meas module. This module detects the transition of data_in by averaging over several cycles of clk_160. The phase difference between data_av and clk_samp is averaged over time, and supplied to the data_sync module. (c) shows a timing diagram for the data_sync module. This module delays data_in by the number of pulses of clk_160 required to ensure that data_out is approximately aligned with the sampling clock.
for three consecutive 160 MHz clock cycles.

The input data level is recorded on the positive edge of the clock, and summed with the previous four values. The output is held low if $sum \in \{0, 1, 2\}$, and high if $sum \in \{3, 4, 5\}$ (see A.1.4.2). The phase delay introduced by $wvfr_{\text{avg}}$ is simply added to the DLL's input phase delay and eliminated.

5.1.2 Measuring the Offset

It is important that data is sampled well away from its transition. This is normally accomplished by sampling on the negative edge of the clock. This is a challenge for the TART as there is an unknown phase difference between the clock and the data. In order to synchronise the data with the clock, the phase offset must first be measured.

The $o\text{ffset}_\text{meas}$ module measures the phase delay of the data relative to the sampling clock. It is supplied with clocks at 16.368 MHz and 160 MHz ($\text{clk}_{\text{samp}} & \text{clk}_{160}$, see Figure 5.2). The first matches the frequency of the reference clock used by the radio module. The second ($\approx 10 \times$) is used for phase measurement, and is generated from a 50 MHz clock by the FPGA’s internal Digital Clock Manager (DCM). Multiplying the sampling clock directly would be preferable, but a minimum input frequency of 18 MHz is required by the DCM [39, pg. 134]. A reset line serves to zero internal counters.

All data channels are assumed to have identical phase delay, because they are transmitted over equal length cables. Therefore only the delay between $\text{clk}_{\text{samp}}$ and $\text{data}_{\text{in}}(0)$ is measured. This value is fed to the $\text{data}_\text{sync}$ module by $\text{offset}(3:0)$ as a 4-bit integer.

The positive transitions in the sampling clock and data signals are detected on the positive and negative edges of the 160 MHz clock respectively (see A.1.4.1). A counter ($\text{count}$) is reset on the positive edge of the sampling clock, and is incremented on each positive edge of the 160 MHz clock. The value of $\text{count}$ is recorded on the positive edge on $\text{data}_{\text{in}}(0)$, and represents the current offset.

The offset is averaged over some time to mitigate any effects of noise. The mode is considered the most suitable measure of central tendency as its value is
least likely to be affected by noise.

The frequency with which each offset value occurs is recorded in a histogram, from which the mode is determined. The offset values are expected to lie in the range $1 \rightarrow 10$, given the $10\times$ relationship between clocks. Sixteen bins, implemented as register $\text{histgrm}(15:0)$ are more than sufficient.

The appropriate bin is incremented on the positive data edge, according to the value of the current offset. The value of this bin is then compared to a register (maximum) containing the maximum bin size. If it is larger, the current offset is written to a register (max_bin) representing the current mode. This is in turn output to the data_sync module where it is used to calculate an appropriate delay for the data signals.

The size of register $\text{histgrm}(15:0)$ is set to $2^{\text{rng_bit}}$, and this determines the maximum value of each bin. Incrementing beyond this would set a bin to zero, and cause the mode to be misreported. As a bin is incremented, its value is checked. If it is full, a right bit shift is performed to halve the frequency held in each bin. Blocking assignments within the verilog code allow the increment, check and divide to be executed in one clock cycle. Since every bin is halved, the position of the maximum (and hence the mode) remains unchanged.

### 5.1.3 Aligning the Data with the Sampling Clock

One data_sync module is provided per channel of data, as the functional block diagram in Figure 5.2 shows. Each creates a copy of data_in that is phase delayed by an integer number of 160 MHz clock cycles. The required delay is calculated from the offset(3:0) input.

The counters cnt_hi and cnt_lo are both incremented on the positive edge of the clock, provided data_in has not changed (see Figure A.1.4.2). If a positive edge edge is detected on data_in, cnt_hi is reset while cnt_lo is incremented. The converse is true on the negative edge.

The always@* block detects when either counter reaches a pre-set value (cnt_val), and sets data_out_next accordingly. When cnt_hi reaches cnt_val, data_out_next is set high. Similarly data_out_next is set low when cnt_lo = cnt_val. The
data output transitions occur on the next positive edge of the clock.

The output data will be in phase with the sampling clock within about 1/10 of the period. The DLL introduces significant phase error ($\approx 63$ ns) in the data signal, but this does not impact reliability. The data is sampled at the midpoint of each bit, on the negative edge of the sampling clock. Even at maximum phase error there is a safe margin between the data transition and sampling points - see Figure 5.3a.

5.2 The Control Buffer Finite State Machine

![State diagram of the control buffer finite state machine](image)

**Figure 5.4:** State diagram of the control buffer finite state machine

The control buffer records samples of fixed length from two to eight sources. The sampled data is then sent to the UART for transmission to a PC via an RS232 link. The states of this Finite State Machine (FSM), and the transitions between them are described as follows;
**Idle:** This is the initial state. It is also entered upon system reset, or after all the contents of the memory have been read. Read and Write Pointers are reset to zero. The `SAMP_BEG` signal selects the write state.

**Write:** In this state, the write pointer is advanced on the positive edge of the sample clock. When the memory is full, the FSM changes to the read state.

**Read:** In this state, the contents of the control buffer are written to the UART a block at a time. The read pointer advances on the positive edge of the sampling clock until a `TX_FULL` signal is received from the UART indicating that its FIFO is full. At this point the FSM changes to the read wait state. When the control buffer memory has been fully emptied, the FSM is returned to the idle state.

**Read Wait:** This state is entered from the read state when the UART FIFO is full. When the `TX EMPTY` signal indicates that the UART FIFO has emptied, the FSM returns to the read state.

### 5.2.1 Memory Utilisation and Sample Sizes

The control buffer utilises up to 54 kB of block RAM available on the Xc3s1000 chip. The width of the memory is determined by the number of signals to be simultaneously recorded - between two and eight. The initial implementation used two channels, and the memory was configured as $2 \times 2^{17}$ bits. Eight channels would be configured as $8 \times 2^{15}$ bits. In each case 32 kB of block RAM are used. The use of three or seven channels would be most efficient, and allow the utilisation of 48 kB. The constraint on memory size imposed by the use of block RAM is recognised. Section 6.2 discusses future plans to extend memory capacity by utilising external SDRAM.

### 5.3 The UART

The Universal Asynchronous Receiver Transmitter (UART) module provides an RS232 interface for communication with a computer. The interface provides adequate speed, is readily available, is straightforward to implement. The required
drivers and DB9 connector are available on the Spartan-3 development board used.

The baud rate, number of data bits and stop bits, and the width of the FIFO are defined as parameters. These are set as 115200 kBit/s, 8 bits, 1 bit, and 16 bytes respectively. Existing code for the UART [8] has been modified to suit this application.

The UART module incorporates a baud rate generator, a receive unit, a transmit unit, and two FIFOs. While the facility for full duplex communication is provided, only the transmit functionality is used at this time.

Each word transmitted contains a one bit sample from up to eight input channels, or a two bit sample from up to four channels. This is dictated by the maximum word size in the RS232 standard. Implementing a second RS-232 port would allow eight two bit channels to be transmitted. This could be extended further by sending multiple bytes sample.

At various stages of the TART’s development, only two, three, or four channels have been used. The remaining bits are set by the rand_fill module to prevent framing errors (see § 5.3.1). At present six one-bit channels are used. In this case the rand_fill module is not required, and the unused bits are simply set to zero.

**Figure 5.5:** Register transfer level block diagram of the UART module. Note that the receive section is not used at this stage.
5.3.1 Prevention of Framing Errors

In RS-232 communication, an eight bit byte is typically framed by a low start bit, and a high stop bit. For example the ASCII code for zero is transmitted as 0001100001 including the start and stop bits (shown bold).

Suppose a continuous stream of these zeros is transmitted. If synchronisation was lost, the same data stream could be misinterpreted as the ASCII character commonly used for XON (11_{16}).

\[
\begin{array}{c}
0 \\
00110000 \\
100110000100011000010001100001 \\
00011000100011000110001100011000110001 \\
\end{array}
\]

\text{XON}

In normal circumstances transmission of a continuous stream of identical characters is rare, and framing errors are recovered from quickly. Where transmission of a repeating sequence is likely, precautions should be taken to prevent framing errors.

During early testing, only two of the eight available channels were used. The first six bits of each recorded byte were therefore zero. A value of 48_{10} was added so the data was transmitted as ASCII \{0,1,2\}. This allowed the incoming data to be displayed in a terminal window in a human readable format.

The \texttt{rand\_fill} module (see A.1.1) was introduced to prevent framing error described above. This module utilised a Linear Feedback Shift Register (LFSR) to generate the first six digits. An LFSR can generate a maximal length pseudorandom sequence of period \(2^n - 1\) - 63 in this case. This strategy eliminated long repeated runs, proved effective in the prevention of framing errors.

5.4 Data Collection and Management.

The data transmitted by the radio modules must be collected and stored for later analysis. This is facilitated by a Spartan-3 FPGA development board and a Raspberry Pi computer. The latter executes Python code to encapsulate the data and save it to a remote network drive.
5.4.1 Recording an Observation

The process of data sampling and storage is initiated at regular intervals by a cron job which executes a shell script. The script in turn executes the `sig_acquire.py` (see B.1). The functionality of this python code is outlined below;

1. Reads command line arguments and configuration file.
2. Performs necessary co-ordinate transformations.
4. Opens and configures the RS-232 port.
5. Generates an accurate UTC time stamp using NTP.
6. Initiates data sampling on the FPGA.
7. Reads data from RS-232 port as a string.
8. Converts the string to an integer array.
9. Creates an observation object from integer array and configuration data.
10. Saves the observation object to a (cpickle) file.

The structure of the observation object is shown in Figure 5.6.

**Required Preparation.** For observations to be recorded a cron job must be set up, a shell script prepared, and the serial port opened. The crontab file is edited by issuing the command `crontab -e`

```
#/bin/sh
TART=/home/projects/TART/software/operation
/usr/bin/python ${TART}/sig_acquire.py --config-file ${TART}/telescope_config.json --data-directory /freenas/telescope/data/
```

The following shell script runs `sig_acquire.py`. It is automatically executed by a cron job at one minute intervals.

```
#/bin/sh
TART=/home/projects/TART/software/operation
/usr/bin/python $(TART)/sig_acquire.py --config-file $(TART)/telescope_config.json --data-directory /freenas/telescope/data/
```

The required serial port is open and set using the `stty` command;

```
stty -F /dev/ttyUSB0 115200 -cstopb
```
The sampled data is stored as a vector of 8-bit integers. Each bit represents a 1-bit sample from a particular channel. When less than 8 channels are connected, `num_ant` allows invalid data to be ignored. The position of antennas and time of observation is also recorded. This information may be recorded the `save` method. A vector of bi-polar binary data for a given channel may recovered be `get_antenna method`. Recovering data on demand reduces file size considerably.

In the example a USB RS-232 device is opened. The baud rate and the number of stop bits are set to 115200 and one respectively.
Chapter 6

Results and Future Work

This chapter reflects on the progress to date, and the performance constraints of the TART’s current specification. A future direction for the project is discussed. This includes further hardware development, and installation at a remote location.

6.1 Results

The clock conditioning, data transmission and radio sampling systems were tested independently. This functionality was separated onto two separate boards in an early prototype of the radio module.

**Clock Conditioning**  The jitter cleaner was able to lock to the clock signal provided to it via the CAT-6 cable. There was a clear improvement in the amount of jitter on the output (see Figure 3.7). The amount of residual phase offset appears to be stable over time. This is indicated by repeatability of the visibility functions over a number of subsequent days of recording.

**Data Transmission**  During preliminary testing of the data transmission system, the jitter cleaner was removed, and its signal replaced by a pseudo-random deterministic code. The latter was produced by a Gold code generator implemented in an FPGA [24]. Comparison of the recorded data with that expected for the Gold code sequence verified correct operation.
**Recorded Spectrum**  The verification of a reliable data transmission and recording system allowed testing of the radio front end. Examining the spectrum of the recorded radio data provided an indication of correct operation. A spectrum for three radio modules is shown in Figure 6.1.

The recorded spectrum tends to demonstrate a centre frequency of 4 MHz, and a bandwidth of 2.5 MHz as expected (see Table 2.1).

![Figure 6.1: Spectrum plots for three channels of 1-bit sampled data. Note that the fifth order Butterworth filter implemented in the radio front end has a centre frequency of about 4 MHz, and a bandwidth of 2.5 MHz. No shielding is installed on either the radio front end, or the jitter cleaner.](image)

**Artificial Source.**  The response of the TART to a single point source was investigated with the aid an external radio noise source transmitting in the GPS L1 band. Two antennas were set up a distance of 0.5 m apart, and an arc of radius 5 m scribed about their centre. Data was recorded as the noise source was moved.
around the circumference in $5^\circ$ increments. The recorded visibility is shown in Figure 6.2 super-imposed on the fringe function (see Equation 3.1 and Figure 3.1). The measured visibilities cycle between positive and negative with the expected frequency.

![Graph showing visibility and fringe function](image)

**Figure 6.2:** Visibility recorded for artificial source located at a distance of 5m from by two antennas located 0.5m apart.

**Astronomical Sources.** To test the response of the TART to astronomical sources, four antennas numbered 0 to 3 were placed at the locations described below:
Antenna N = \{x, y, z\}
Antenna 0 = \{0, 0, 0\}
Antenna 1 = \{0, 0.5, 0\}
Antenna 2 = \{-1.0, 6.5, -0.35\}
Antenna 3 = \{-1.0, 7.0, -0.35\}

The orthogonal co-ordinate system is right handed with the X axis aligned on a bearing of 112°. The visibilities plotted in Figure 6.3 were recorded over the course of a day, and are as expected for a number point tracking across the sky.

Inspecting the plot, note the frequency of the visibility function \(V(0, 2)\) is considerably higher than \(V(0, 1)\). This is as expected and reflects the fact antenna pair (0, 1) has a baseline of about one thirteenth of (0, 2). Note also that \(V(2, 3)\) has a lower frequency than \(V(0, 2)\). The antenna pair (2, 3) has a baseline of 19 cm projected on to the x-axis, compared to (0, 1) which is 46 cm. Recall that the length of the baseline determines the resolving power of the instrument.
Figure 6.3: The signal from each antennas (four - numbered 0 to 3) is correlated with the signal from every other antenna. The resulting complex visibilities are plotted above. The complex visibility $V_{2,3}$ represents the correlated signal from antennas 2 and 3. The phase is the lower green trace, while the magnitude is the upper blue one. The bottom most pane shows a preliminary measure for transient detection. The trace represents the product of the derivative of the visibility of each antenna pair.
6.2 Future Work

The radio interferometer described in this thesis was conceived as proof of concept device. Limitations include;

- A maximum of eight antennas.
- A buffer implemented in FPGA block RAM.
- Data transfer by RS-232 at a maximum of 115.4 kb/s baud.

Goals for future generations of the TART include;

- Improving the sensitivity.
- Improving spatial frequency $(U,V)$ coverage.
- Improving clock Distribution System.
- Improving FPGA programming interface.
- Reduction of interference and atmospheric attenuation.

**Improve Sensitivity** The current sensitivity of the TART is limited. At present, only one 4 ms observation is recorded per minute from six antennas of 90° HPBW. The sensitivity may be improved by increasing the duration of each observation, or increasing the antenna gain by decreasing the HPBW (see §2.5).

The observation time may be extended by increasing the buffer size. This would require the use of external SDRAM. In order to maintain the observation frequency, the data throughput would also need to be increased. At present the sampled signal is buffered in the FPGA’s internal block RAM for transmission by RS-232 at a data rate of 92.16 kb/s. Real time data transfer would remove any restriction on the duration of each observation. This would require a FIFO buffer, and a very fast data transfer protocol.

Increasing the antenna gain increases the signal to noise ratio $\mathcal{R}_{an}$ by both increasing the antenna temperature $T_a$, and reduces the system temperature $T_s$ (see Equation 1.2). The effect of antenna gain on $T_a$ and $T_s$ is discussed in Sections 2.5 and 2.3.3 respectively.
**Improve Spatial Frequency (U,V) Coverage.**  The fact that the TART only records one sample per minute from a maximum of eight antennas limits the spatial frequency coverage, and the quality of any image that may potentially be generated.

For $N$ antennas spread over the surface of the earth, at any instant there are $^N\!C_2$ spatial frequency components or samples in the $(U,V)$ plane. The Earth rotation synthesis technique allows the coverage of the $(U,V)$ plane to be extended. This is because the baseline vector seen from the source changes continuously as the Earth rotates [3, pg.34].

Increasing the number of antennas, and the frequency with which observations are recorded would allow improved $(U,V)$ coverage. This will require more RAM for buffering, and faster data transfer.

**Improve Clock Distribution System**  The manner in which clock signals are supplied to the FPGA does not currently represent best design practice. The current implementation also relies on using identical signal paths to maintain phase coherence between interferometer channels. In practice this can not be assumed (see §3.3.2).

The FPGA’s global clock input (GCK4) is supplied with a 16.368 MHz signal via a pin on the development board’s expansion connector. The capacitance in this line caused reliability issues early on. Placing the oscillator next to the FPGA would address this issue.

A $10\times$ oversampling clock required by the DLL used to synchronise the incoming data with the sampling clock. Unfortunately, this had to be generated from a 50 MHz oscillator on the development board, rather than 16.368 MHz supplied. This is because the Spartan-3’s DCM requires a minimum input frequency of 18 MHz. Upgrading to an FPGA with a more capable DCM would allow a single clock domain, and possibly provide a PLL to condition the incoming clock signal.

Making the suggested improvements to FPGA clocking introduces the possibility of supplying a clock signal to each radio module directly from the FPGA. The capabilities of the current DLL could be extended to provide closed loop control
of the clock phase supplied to each radio module. Access to the FPGA’s DCM would help facilitate this.

**Improve FPGA Programming Interface**  The Spartan-3 development board used by the TART is programmed by Xilinx’s proprietary USB platform cable USB II. Driver and software support for this device is poor, especially in Linux. Future Technology Devices International (FTDI) provide USB to JTAG bridges which are inexpensive, and have excellent driver support. Their use would allow FPGA to be re-programmed remotely in the field.

**Reduce Interference and Atmospheric Attenuation.** The TART requires a location that has low atmospheric water vapour, and is radio quiet. Presently, the TART is located on the roof of the Science III building at the University of Otago. While this has permitted ease of access during the hardware testing phase, it is less than ideal for collecting astronomical data. A future version of the TART will be installed at a high altitude remote location.

### 6.2.1 Remote Deployment

The proposed site for the TART is located near Sutherland’s Peak on the Benmore ranges, in New Zealand’s Mackenzie Basin. This area was gifted to University of Otago at its formation in 1869. The site itself does not form part of any leasehold agreement. Human settlement in the immediate vicinity is sparse. Timaru, the nearest significant town, is located 90km due east and is separated by numerous mountain ranges. The nearby Waitaki hydro-electric project is very unlikely to be a source of interference in the bands of interest.

The site is located some 1800 m above sea level, and is very dry. The relative humidity is often less that 10%, and precipitation is minimal - usually in the form of snow. Wind loading is an important consideration for any structure located here. New Zealand’s highest recorded wind gust of 250 km/h was measured at nearby Mount John Observatory on 18th April 1970. The TART site is nearly 800 m higher and considerably more exposed!
Figure 6.4: The proposed site is a plateau at 1800 m near Sutherlands Peak. This is located on the Benmore Range, in New Zealand’s Mackenzie Basin. Access to this University of Otago owned land may be gained on foot, or by helicopter. The structures at the right are the remains of a radio relay station.
Access Access to within a short distance of the site is possible by four wheel drive. The tracks cross adjoining farm land, and permission for access must be sought from land owners. While the site appears rugged, it is actually very fragile ecologically, and protecting it from damage is a high priority. The plan is to restrict vehicle access except in an emergency. It is envisioned that access will ordinarily be on foot. Access to the site by helicopter is straightforward and relatively inexpensive, and is suitable for larger items such as building materials. Flight time from Pukaki airport in Twizel is less than ten minutes.

Required Facilities An existing hut on the site once housed a radio relay station for the Waitaki hydro project. It was built in the 1950s, and is currently in a dilapidated condition. In the short term the existing hut will be braced, and made weather tight. The consenting process for a planned replacement is likely to take some time, given Benmore Range’s status as an Outstanding Landscape of National Importance.

Electrical power will be provided by solar panels affixed to the north wall of the hut. This location allows them to be kept clear of snow in the wintertime. High winds speeds at the locality tend to preclude the use of turbines.

A permanent internet connection is required to allow remote data recovery. It is hoped that a wireless link can be established with Mt John Observatory, located within line of sight. Mt John is connected to the ultra high speed Kiwi Advanced Research and Education Network (KAREN).
6.3 Conclusion

This project set out to demonstrate the viability of an L1 band radio interferometer using consumer electronic components. Such a device has been constructed and operated reliably over a period of several months. It has performed according to its design specification. Results indicate detection of extra-terrestrial signals, and stable interferometer phase coherence.

Currently the performance is limited by the quantity and design of antennas, and the frequency and duration of each observation. These factors affect the spatial frequency \((U,V)\) coverage, the sensitivity, and the probability of recording a transient event. Addressing these issues will allow the TART to transition from a proof of concept device to a useful instrument for algorithm development.
Bibliography


[26] Silicon Laboratories Inc., 400 West Cesar Chavez, Austin, TX 78701. AN591 - Crystal selection fo the Si5315, Si 5317, and other Si53XX Any-Frequency Jitter Attenuationing Clocks, 0.4 edition, March 2012.


[33] TXC corporation. *SMD Temperature Compensated Crystal Oscillators 3.2 by 2.5 by 1.0 mm Q Series*.


Appendix A

Verilog Code for base station

A.1 sig_to_rs232.v

module sig_to_rs232
 (#
 parameter DATA_WIDTH = 6, // Width of data in control buffer
 parameter ADDR_WIDTH = 16 // Width of address bus
 // Check memory usage for address bus width and data width
 // bits used / num bits in kB
 // 6 * 2^16 / (8*2^10) = 48 < 54kB so ok
 )

 input wire clock, // On board 50MHz clock
 input wire clk_16_Meg_in, // External 16.368 MHz ref clock
 input wire reset, // Manual reset button
 input wire samp_beg_sw, // External switch to initiate sampling
 input wire [7:0] dt_sign, // signal input - sign bit
 input wire [7:0] dt_mag, // signal input - magnitude bit (unused)
 input wire rx, // RS-232 receive line. Sampling initiated on high
 output wire tx, // External transmit line of the UART
 output wire [7:0] data_leds // Make output to light leds
);

// Parameters for generating random fill data (Prevent RS232 framing errors)
localparam fill_bits = 8 - DATA_WIDTH; // # of fill bits of rnd data reqd
localparam fill_taps = 2'b11; // Tapping for LFSR for rand fill data
localparam fill_lfsr_init = 2'b11; // Initial condition of LFSR (ADVANCED BEFORE READ !!!!)

// Inter-connect
wire clk_16_Meg; // Ref clock for data sampling buffered (not phase corrected)
wire clk_160_Meg; // Provide approx 10x oversampling for DLL
wire tx_full; // Indicates UART transmit FIFO is full
wire tx_empty; // Indicates UART transmit FIFO is empty
wire wr_uart; // A write enable signal for the UART transmit FIFO
wire [7:0] w_data; // Outgoing data to UART
wire [7:0] r_data; // Incoming data from UART
wire [DATA_WIDTH-1:0] din; // Data from DLL to ctrl buffer
wire [1:0] state_reg; // State of the control buffer
wire [DATA_WIDTH-1:0] dout; // Data from control buffer to UART
wire [7:0] dt_sign_pad; // Data padded with zeros if < 8 channels used
wire samp_beg; // Sampling may be initiated manually, or by a high on the rx line
assign samp_beg = rx; // (samp_beg_sw || rx) ? 1'b1 : 1'b0;
// Inputs for unused channels set to zero. Inputs to 4 LSB's
assign dt_sign_pad = {2'b00, dt_sign[DATA_WIDTH-1:0]};

// Assign unused bits as PRN to prevent framing error
rand_fill #(.bits(fill_bits), .taps(fill_taps), .lfsr_init(fill_lfsr_init))
rand_fill_inst (.clk_samp(clk_16_Meg), .reset(reset), .wr_uart(wr_uart), .tx_full(tx_full), .data(w_data[7:DATA_WIDTH]));

// BUFG: Global Clock Buffer
BUFG BUFG_inst (.O(clk_16_Meg), // Clock buffer output
 .I(clk_16_Meg_in) // Clock buffer input
);

// This module produces a 160MHz clock for PLL which aligns data to clock
dcm_160_Meg dcm_160_Meg_inst(.CLKIN_IN(clock), .CLKFX_OUT(clk_160_Meg), .CLKIN_IBUFG_OUT());

// DLL aligns data with clock to ensure that data is not sampled on an edge
delay_lock_loop #(.DATA_WIDTH(DATA_WIDTH))
delay_lock_loop_inst (.clk_160(clk_160_Meg), .clk_samp(clk_16_Meg), .data_in(dt_sign_pad), .reset(reset), .data_out(din));

// User definable parameters for the UART
// sb_tick = 16 for one stop bit, dbit = 8 for 8 data bits,
// fifo_w = 4 for 2^4 = 16 fifo bytes
// {baud, dvsr, dvsr_bit} as follows {{115200,27,5}, {57600,54,6},
// {9600,326,9}, {19200,163,8}, {38400,487,9},
uart #(.DBIT(8), .SB_TICK(16), .DVSR(9), .DVSR_BIT(5), .FIFO_W(4))
uart_inst (.clk(clk_16_Meg), .reset(reset), .rd_uart(), .wr_uart(wr_uart),
 .rx(), .w_data(w_data), .tx_empty(tx_empty), .tx_full(tx_full), .rx_empty(),
 .tx(tx), .r_data(r_data), .tx_fifo_out(data_leds));
// Instantiate buffer
control_buffer #(ADDR_WIDTH, DATA_WIDTH)
    buffer_inst (.clk_samp(clk_16_Meg), .reset(reset), .din(din),
        .samp_beg(samp_beg), .tx_empty(tx_empty), .tx_full(tx_full),
        .dout(w_data[DATA_WIDTH-1:0]), .wr_uart(wr_uart), .state_reg(state_reg));
endmodule

A.1.1 rand_fill.v

module rand_fill
#(
    parameter bits = 3'd6,
    parameter taps = 6'b100000, // Tapped at bit 6 & 5
    parameter lfsr_init = 6'b111111 // (Advanced before read)
)
(
    input clk_samp,
    input reset,
    input wr_uart,
    input tx_full,
    output [bits-1:0] data
);

wire clock;
reg enable = 1'b0;

assign clock = (wr_uart & ~ tx_full) ? clk_samp : 1'b0;
// Advance LFSR only when ctrl buff in read mode
// Takes clock cycle for control buffer to read tx_full and output wr_uart
// Avoid lag by also using tx_full signal as well

// Instantiate LFSR #1
// Note the LFSR is updated on the positive edge of the sampling clock
// and its state is read on the negative edge of the sampling clock (by UART)
fibonacci_lfsr #(.bits(bits), .taps(taps), .init_state(lfsr_init))
    lfsr_1(.clock(clock), .reset(reset), .signal_out(data));
endmodule

A.1.1.1 fibonacci_lfsr.v

module fibonacci_lfsr


A.1.2  dcm_160_Meg.v

module dcm_160_Meg(CLKIN_IN, CLKFX_OUT, CLKIN_IBUFG_OUT);

input CLKIN_IN;
output CLKFX_OUT;
output CLKIN_IBUFG_OUT;

wire CLKFX_BUF;
wire CLKIN_IBUFG;
wire GND_BIT;

assign GND_BIT = 0;
assign CLKIN_IBUFG_OUT = CLKIN_IBUFG;
BUFG CLKFX_BUF_INST (.I(CLKFX_BUF), .O(CLKFX_OUT));
IBUFG CLKIN_IBUFG_INST (.I(CLKIN_IN), .O(CLKIN_IBUFG));
DCM #( .CLK_FEEDBACK("NONE"), .CLKDV_DIVIDE(2.0), .CLKFX_DIVIDE(6), .CLKFX_MULTIPLY(16),
A.1.3 delay_lock_loop.v

module delay_lock_loop
#
 parameter rng_bit = 3'd4, // Maximum offset 160/16.368 = 10 approx allow 2^4 = 16
 parameter DATA_WIDTH = 2
)
(
 input clk_160,
 input clk_samp,
 input [7:0] data_in, // CHANGE - ONE DATA SYNC REQUIRED FOR EACH CHANNEL OF DATA
 input reset,
 output [DATA_WIDTH-1:0] data_out
);

wire [3:0] offset;
wire [7:0] sync_out;

assign data_out = sync_out[DATA_WIDTH-1:0];

// Instantiate module to measure phase offset between data and sampling clock
offset_meas #(rng_bit(rng_bit), .max_off(4'd8))
 offset_meas_inst(clk_160(clk_160), .clk_samp(clk_samp), .data(data_in[0]), .reset(reset), .
 offset(offset));

// Instantiate module to regenerate a sampling clock of the required phase offset.
// Note - he offset measured for data_in[0] is assumed to be identical for all channels
data_sync #(rng_bit(rng_bit)) data_sync_inst_0 (.clk_160(clk_160), .reset(reset), .data_in(.
 data_in[0]), .offset(offset), .data_out(sync_out[0]));
data_sync #(rng_bit) data_sync_inst_1 (.clk_160(clk_160), .reset(reset), .data_in(data_in[1]), .offset(offset), .data_out(sync_out[1]));

data_sync #(rng_bit) data_sync_inst_2 (.clk_160(clk_160), .reset(reset), .data_in(data_in[2]), .offset(offset), .data_out(sync_out[2]));

data_sync #(rng_bit) data_sync_inst_3 (.clk_160(clk_160), .reset(reset), .data_in(data_in[3]), .offset(offset), .data_out(sync_out[3]));

data_sync #(rng_bit) data_sync_inst_4 (.clk_160(clk_160), .reset(reset), .data_in(data_in[4]), .offset(offset), .data_out(sync_out[4]));

data_sync #(rng_bit) data_sync_inst_5 (.clk_160(clk_160), .reset(reset), .data_in(data_in[5]), .offset(offset), .data_out(sync_out[5]));

data_sync #(rng_bit) data_sync_inst_6 (.clk_160(clk_160), .reset(reset), .data_in(data_in[6]), .offset(offset), .data_out(sync_out[6]));

data_sync #(rng_bit) data_sync_inst_7 (.clk_160(clk_160), .reset(reset), .data_in(data_in[7]), .offset(offset), .data_out(sync_out[7]));

```
A.1.4  wvfm_avg.v

module wvfm_avg
(
    input clk_fast,
    input sig_in,
    output wire sig_avg
);

    reg [4:0] hist = 4'b0;
    reg [2:0] sum = 3'b0;

always @ (posedge clk_fast)
begin
    hist <= {hist[3:0], sig_in};
end

always @ *
begin
end

assign sig_avg = sum[2] | (sum[1] & sum[0]); // high if sum 3, 5 or 5. 0,1,2 low
```

116
A.1.4.1 offset_meas.v

module offset_meas
#
  parameter rng_bit = 3'd4, // Maximum offset 160/16.368 = 10 approx allow 2^4 = 16
  parameter max_off = 4'd15, // If this value is exceed, offset set to zero (default not used)
  parameter max_val_bit = 4'd8 //The number of bits determines the maximum value that can be stored
   in each bin

  input clk_160, // A fast clock signal for sampling data and clk_samp and accurately determining
        the offset
  input clk_samp, // The input sampling clock with correct period that requires phase correction
  input data, // The data that the sampling clock is to be phase locked to
  input reset,
  output wire [3:0] offset // Current offset - ie the histogram bin with maximum entries (Mode)
);

reg clk_samp_prev = 1'b0; // Used to detect signal edge
reg data_prev = 1'b0; // Used to detect signal edge
reg [3:0] count = 0; // Count # fast clk cycles between data & clk_samp edge
reg [max_val_bit-1:0] histgrm [((2**rng_bit)-1):0]; // one bin for each possible offset count
   up to 1023
reg [max_val_bit-1:0] maximum = 0; // The current maximum value
reg [rng_bit-1:0] max_bin = 0;
wire data_av;

integer i; // Used for for loop to halve bin values
initial
  begin
    for(i = 2**rng_bit-1; i>=0; i=i-1)
      begin
        histgrm[i] <= {max_val_bit{1'b0}}; // clear each histogram bin in turn
      end
  end

// average incoming signal to mitigate effects of noise
wvfm_avg wvfm_avg_inst_clk(.clk_fast(clk_160), .sig_in(data), .sig_avg(data_av));
// 23/1/2013 – Amended to update histogram on the negative edge of the data

// Avoid glitches when data edge occurs within one fast clock cycle of the (averaged) sampling
// clock
assign offset = (max_bin <= max_off) ? max_bin : {rng_bit{1'b0}}; // If max offset is exceeded
  set offset to zero

always @posedge clk_160
begin
  if(reset)
    begin
      count <= 0;
    end
  else
    begin
      clk_samp_prev <= clk_samp; // Used for detection of sampling clock edge
      data_prev <= data_av; // Used for detection of data edge
      count <= count + 1; // Update counter used to measure offset
    end
  end

always @negedge clk_160
begin
  if(reset)
    begin
      max_bin = 0;
      // Clear all the values of the histogram
      for(i = 2**rng_bit-1; i>=0; i=i-1)
        begin
          histgrm[i] = {max_val_bit{1'b0}}; // clear each histogram bin in turn
        end
    end
  else
    begin
      if (data_av & ~data_prev) // Detect ONLY POSITIVE data edge
        begin
          histgrm[count] = histgrm[count] + 1'b1; // Increment appropriate bin of histogram
          // depending on current offset
          // Update value and position of maximum bin if this has changed
          // #############################################################
          if (histgrm[count] >= maximum)
            begin

```
A.1.4.2 data_sync.v

// This module synchronizes data to a reference clock
module data_sync
#(
    parameter rng_bit = 3'd4, // Maximum offset 160/16.368 = 10 approx allow 2^4 = 16
    num_steps = 4'd10 // use to calculate data delay)
(
    input clk_160,
    input reset,
    input data_in,
    input [rng_bit-1:0] offset,
    input [3:0] offset,
    output reg data_out = 0
);

reg [15:0] cnt_hi = 0; // Increased from 8 to 16 bit
reg [15:0] cnt_lo = 0; // Increased from 8 to 16 bit
reg data_in_prev = 0;
reg data_out_next = 0;
wire [3:0] cnt_val;
wire data_in_av;

assign cnt_val = num_steps - offset - 1; // Fudge factor of 1 to account for delay

max_bin = histgrm[count]; // New position of the maximum bin. This is the mode, and the
current offset value.
end

// Just before overflow occurs in largest bin, halve all values of the histogram
if(maximum == {4{1'b1}})
begin
  for(i= 2**rng_bit-1; i>=0; i=i-1)
  begin
    histgrm[i] = histgrm[i]>>1; //rightshift one bit = equates to division by 2 (LSB lost)
  end
  maximum = maximum >> 1;
end
end

endmodule
// average incoming signal to mitigate effects of noise
wvfm_avg wvfm_avg_inst_clk(.clk_fast(clk_160), .sig_in(data_in), .sig_avg(data_in_av));

always @ (posedge clk_160)
begin
    if (reset) // Reset all registers
    begin
        data_out <= 0;
        cnt_hi <= 0;
        cnt_lo <= 0;
        data_in_prev <= 0;
    end
    else
    begin
        data_out <= data_out_next; // Change state of output clock
        data_in_prev <= data_in_av; // for detection of sampling clock edge
        if (data_in_av != data_in_prev) // Detect clock edge - zero appropriate counter
        begin
            if (data_in_av & ~data_in_prev) // Posedge data_in
            begin
                cnt_hi <= 0; // Reset counter from posedge data_in
                cnt_lo <= cnt_lo + 1; // Increment counter from negedge data_in
            end
            else // Negedge data_in
            begin
                cnt_lo <= 0; // Reset count from negedge clk_samp
                cnt_hi <= cnt_hi + 1; // Increment counter from posedge data_in
            end
        end
        else // Not at a clock edge
        begin // Increment counters for pos and negedge of data_in
            cnt_hi <= cnt_hi + 1;
            cnt_lo <= cnt_lo + 1;
        end
    end
end

always @* // change output clock after req'd no clk cycles from edge of sampling clock
begin
    data_out_next <= data_out; // Default - no change
    if (cnt_hi == cnt_val)
        data_out_next <= 1;
    else if (cnt_lo == cnt_val)
        data_out_next <= 0;
end
endmodule
module uart
#
//Default setting:
// 19,200 baud, 8 data bits, 1 stop bit, 2^2 FIFO
parameter DBIT = 8, // # data bits
parameter SB_TICK = 16, // # ticks for stop bits,
// 16/24/32 for 1/1.5/2 bits
parameter DVSR = 163, // baud rate divisor
// DVSR = 50M/16*baud rate)
parameter DVSR_BIT = 8, // # bits of DVSR
parameter FIFO_W = 3 // # addr bits of FIFO
// # words in FIFO=2^FIFO_W

input wire clk, reset,
input wire rd_uart, wr_uart, rx,
input wire [7:0] w_data,
output wire tx_empty,
output wire tx_full, rx_empty, tx,
output wire [7:0] r_data,
output wire [7:0] tx_fifo_out
);
// signal declaration
wire tick, rx_done_tick, tx_done_tick;
wire tx_fifo_not_empty;
wire [7:0] rx_data_out;

//body
mod_m_counter #(M(DVSR), N(DVSR_BIT)) baud_gen_unit
(.clk(clk), .reset(reset), .q(), .max_tick(tick));

uart_rx #(DBIT(DBIT), SB_TICK(SB_TICK)) uart_rx_unit
(.clk(clk), .reset(reset), .rx(rx), .s_tick(tick),
 .rx_done_tick(rx_done_tick), .dout(rx_data_out));

fifo #(B(DBIT), W(FIFO_W)) fifo_rx_unit
(.clk(clk), .reset(reset), .rd(rd_uart),
 .wr(rx_done_tick), .w_data(rx_data_out),
 .empty(rx_empty), .full(), .r_data(r_data));

fifo #(B(DBIT), W(FIFO_W)) fifo_tx_unit
(.clk(clk), .reset(reset), .rd(tx_done_tick),
 .wr(wr_uart), .w_data(w_data), .empty(tx_empty),
 .full(tx_full), .r_data(tx_fifo_out));
A.1.5.1 mod_m_counter.v

module mod_m_counter
#(
    parameter N=4, // number of bits in counter
    parameter M = 10 // mod-M
)
(
    input wire clk, reset,
    output wire max_tick,
    output wire [N-1:0] q
);

    // signal declaration
    reg [N-1:0] r_reg;
    wire [N-1:0] r_next;

    // body
    // register
    always @(posedge clk, posedge reset)
    if (reset)
        r_reg <= 0;
    else
        r_reg <= r_next;

    // next-state logic
    assign r_next = (r_reg==(M-1)) ? 0 : r_reg + 1;

    // output logic
    assign q = r_reg;
    assign max_tick = (r_reg==(M-1)) ? 1'b1 : 1'b0;
endmodule

A.1.5.2 uart_rx.v
module uart_rx
  #(
    parameter DBIT = 8, // # data bits
    parameter SB_TICK = 16 // # ticks for stop bits
  )
  (input wire clk, reset,
    input wire rx, s_tick,
    output reg rx_done_tick,
    output wire [7:0] dout
  );

  // symbolic state declaration
  localparam [1:0]
    idle = 2'b00,
    start = 2'b01,
    data = 2'b10,
    stop = 2'b11;

  // signal declaration
  reg [1:0] state_reg, state_next;
  reg [3:0] s_reg, s_next;
  reg [2:0] n_reg, n_next;
  reg [7:0] b_reg, b_next;

  // body
  // FSM state & data registers
  always @(posedge clk, posedge reset)
  if (reset)
    begin
      state_reg <= idle;
      s_reg <= 0;
      n_reg <= 0;
      b_reg <= 0;
    end
  else
    begin
      state_reg <= state_next;
      s_reg <= s_next;
      n_reg <= n_next;
      b_reg <= b_next;
    end

  // FSM next-state logic
  always @*
  begin
    state_next = state_reg;
rx_done_tick = 1'b0;
s_next = s_reg;
n_next = n_reg;
b_next = b_reg;
case (state_reg)
    idle:
        if(`rx)
            begin
                state_next = start;
                s_next = 0;
            end
    start:
        if (s_tick)
            if (s_reg==7)
                begin
                    state_next = data;
                    s_next = 0;
                    n_next = 0;
                end
            else
                s_next = s_reg + 1;
    data:
        if (s_tick)
            if (s_reg==15)
                begin
                    s_next = 0;
                    b_next = {rx, b_reg[7:1]};
                    if (n_reg==(DBIT-1))
                        state_next = stop ;
                    else
                        n_next = n_reg + 1;
                end
            else
                s_next = s_reg + 1;
    stop:
        if (s_tick)
            if (s_reg==(SB_TICK-1))
                begin
                    state_next = idle;
                    rx_done_tick = 1'b1;
                end
            else
                s_next = s_reg + 1;
endcase
end

// output
module fifo
#
parameter B=8; // number of bits in a word
parameter W=4; // number of address bits
#
(input wire clk, reset,
input wire rd, wr,
input wire [B-1:0] w_data,
output wire empty, full,
output wire [B-1:0] r_data); //signal declaration

reg [B-1:0] array_reg [2**W-1:0]; //register array
reg [W-1:0] w_ptr_reg, w_ptr_next, w_ptr_succ;
reg [W-1:0] r_ptr_reg, r_ptr_next, r_ptr_succ;
reg full_reg, empty_reg, full_next, empty_next;
wire wr_en;

// body
// register file write operation
always @(posedge clk)
    if (wr_en)
        array_reg[w_ptr_reg] <= w_data;
// register file read operation
assign r_data = array_reg[r_ptr_reg];
// write enabled only when FIFO is not full
assign wr_en = wr & ~full_reg;

//fifo control logic
//register for read and write pointers
always @(posedge clk, posedge reset)
    if (reset)
        begin
            w_ptr_reg <= 0;
            r_ptr_reg <= 0;
            full_reg <= 1'b0;
            empty_reg <= 1'b1;
        end
else
begin
    w_ptr_reg <= w_ptr_next;
    r_ptr_reg <= r_ptr_next;
    full_reg <= full_next;
    empty_reg <= empty_next;
end

// next-state logic for read and write pointers
always @*
begin
// successive pointer values
    w_ptr_succ = w_ptr_reg + 1;
    r_ptr_succ = r_ptr_reg + 1;
// default: keep old values
    w_ptr_next = w_ptr_reg;
    r_ptr_next = r_ptr_reg;
    full_next = full_reg;
    empty_next = empty_reg;
case ({wr, rd})
// 2'b00: no op
  2'b00: //read
    if (~empty_reg) // not empty
      begin
        r_ptr_next = r_ptr_succ;
        full_next = 1'b0;
        if (r_ptr_succ==w_ptr_reg)
          empty_next = 1'b1;
      end
  2'b01: //write
    if (~full_reg) // not full
      begin
        w_ptr_next = w_ptr_succ;
        empty_next = 1'b0;
        if (w_ptr_succ==r_ptr_reg)
          full_next = 1'b1;
      end
  2'b10: //write and read
      begin
        w_ptr_next = w_ptr_succ;
        r_ptr_next = r_ptr_succ;
      end
endcase
end

// output
assign full = full_reg;
module uart.tx
#
parameter DBIT = 8, // # data bits
parameter SB_TICK = 16 // # ticks for stop bits
#
input wire clk, reset,
input wire tx_start, s_tick,
input wire [7:0] din,
output reg tx_done_tick,
output wire tx
);

// symbolic state declaration
localparam [1:0]
  idle = 2'b00,
  start = 2'b01,
  data = 2'b10,
  stop = 2'b11;

// signal declaration
reg [1:0] state_reg, state_next;
reg [3:0] s_reg, s_next;
reg [2:0] n_reg, n_next;
reg [7:0] b_reg, b_next;
reg tx_reg, tx_next;

// body
// FSM state & data registers
always @(posedge clk, posedge reset)
  if (reset)
    begin
      state_reg <= idle;
      s_reg <= 0;
      n_reg <= 0;
      b_reg <= 0;
      tx_reg <= 1'b1;
    end
  else
    begin

A.1.5.4 uart_tx.v
state_reg <= state_next;
s_reg <= s_next;
n_reg <= n_next;
b_reg <= b_next;
tx_reg <= tx_next;
end

// FSMD next-state logic & functional units
always @*
begin
state_next = state_reg;
tx_done_tick = 1'b0;
s_next = s_reg;
n_next = n_reg;
b_next = b_reg;
tx_next = tx_reg;
case (state_reg)
  idle:
    begin
      tx_next = 1'b1;
      if (tx_start)
        begin
          state_next = start;
          s_next = 0;
          b_next = din;
        end
    end
  start:
    begin
      tx_next = 1'b0;
      if (s_tick)
        if (s_reg==15)
          begin
            state_next = data;
            s_next = 0;
            n_next = 0;
          end
        else
          s_next = s_reg + 1;
    end
  data:
    begin
      tx_next = b_reg[0];
      if (s_tick)
        if (s_reg==15)
          begin
            s_next = 0;
      end
end
b_next = b_reg >> 1 ;
if (n_reg==(DBIT-1))
state_next = stop;
else
n_next = n_reg + 1;
end
else
s_next = s_reg + 1;
end

endcase

end

module control_buffer
#(
    parameter ADDR_WIDTH = 6,
    parameter DATA_WIDTH = 8
)
(
input clk_samp, // Sampling clock at 16.368MHz nom
input reset, // Global reset
input wire [DATA_WIDTH-1:0] din, // Data in one channel per antenna
input wire samp_beg, // Begin sampling data to buffer
input wire tx_empty, // UART buffer empty
input wire tx_full, // UART buffer full
output wire [DATA_WIDTH-1:0] dout, // Data_out (to UART)
output reg wr_uart, // Write enable signal for UART
output reg [1:0] state_reg // State register
);

endmodule

A.1.6 control_buffer.v
// Signals
wire we; // Write enable
reg [1:0] state_next = 2'b00;
reg [ADDR_WIDTH : 0] w_ptr, r_ptr;
reg [ADDR_WIDTH : 0] w_ptr_next, r_ptr_next;
reg samp_beg_prev;// = 1'b0;
wire samp_beg_released;

reg wr_uart_next = 1'b0;

// symbolic state declaration
localparam [1:0]
idle = 2'b00,
write = 2'b01,
read = 2'b10,
rd_wait = 2'b11;

// Body
always @(posedge clk_samp, posedge reset)
if (reset)
begin
state_reg <= idle;
// samp_beg_prev <= 1'b0;
wr_uart <=1'b0;
end
else
begin
state_reg <= state_next;
w_ptr <= w_ptr_next;
r_ptr <= r_ptr_next;
wr_uart <= wr_uart_next;
if (samp_beg)
begin
samp_beg_prev <= 1'b1;
end
else
begin
samp_beg_prev <= 1'b0;
end
end

assign samp_beg_released = &{~samp_beg,samp_beg_prev};

always @* begin
state_next = state_reg; // Default - Remain in current state
w_ptr_next = w_ptr; // Default - remain unchanged
r_ptr_next = r_ptr; // Default - remain unchanged

case (state_reg)

idle:
    begin
        // Reset read and write pointers
        w_ptr_next = {ADDR_WIDTH{1'b0}};
        r_ptr_next = {ADDR_WIDTH{1'b0}};
        wr_uart_next = 1'b0;
        if (samp_beg_released) state_next = write;
    end

write:
    begin
        // Memory full, next state read
        if (w_ptr == 2**ADDR_WIDTH - 1 )
            state_next = read;
        // Memory not full increment write pointer
        else
            w_ptr_next = w_ptr + 1'b1;
    end

read:
    begin
        // Memory full, next state idle
        if (r_ptr == 2**ADDR_WIDTH - 1 )
            begin
                state_next = idle;
            end
        // Memory not full increment read pointer
        else
            case (tx_full)
                1'b0:
                    begin
                        r_ptr_next = r_ptr + 1'b1;
                        wr_uart_next =1'b1;
                    end
                1'b1:
                    begin
                        state_next = rd_wait;
                        wr_uart_next = 1'b0;
                    end
            endcase
    end

rd_wait:
begin
  if (tx_empty)
    state_next = read;
  end

  default: state_next = idle;
endcase

assign we = (state_reg == write) ? 1'b1 : 1'b0; // Assign we in write mode

// Instantiate dual port ram

dual_port_ram_sync #(.ADDR_WIDTH(ADDR_WIDTH), .DATA_WIDTH(DATA_WIDTH))
dual_port_ram_sync_inst (.clk(clk_samp), .we(we), .addr_a(w_ptr[ADDR_WIDTH-1:0]),
  .addr_b(r_ptr[ADDR_WIDTH-1:0]), .din_a(din), .dout_a(), .dout_b(dout));
endmodule

A.1.6.1 dual_port_ram_sync.v

module dual_port_ram_sync
#(
  ADDR_WIDTH = 6,
  parameter DATA_WIDTH = 8
)
(
  input wire clk,
  input wire we,
  input wire [ADDR_WIDTH-1:0] addr_a, addr_b,
  input wire [DATA_WIDTH-1:0] din_a,
  output wire [DATA_WIDTH-1:0] dout_a, dout_b
);

  // Signal declaration
  reg [DATA_WIDTH-1:0] ram [2**ADDR_WIDTH-1:0];
  reg [ADDR_WIDTH-1:0] addr_a_reg, addr_b_reg;

  // Body
  always @(posedge clk)
  begin
    if (we) // Write operation
      ram[addr_a] <= din_a;
  end

endmodule
addr_a_reg <= addr_a;
addr_b_reg <= addr_b;
end
// two read operations
assign dout_a = ram[addr_a_reg];
assign dout_b = ram[addr_b_reg];
endmodule

A.1.7 sig_to_rs232.ucf

NET "reset" LOC = L14; // BTN3
NET "tx" LOC = T13; // RS232 tx line goes high to start sampling
NET "samp_beg_sw" LOC = M13; // BTN0 (to manually start sampling)
NET "tx" LOC = R13;

// Output current RS232 byte to leds for debugging
NET "data_leds[0]" LOC = K12; // LD0
NET "data_leds[1]" LOC = P14; // LD1
NET "data_leds[2]" LOC = L12; // LD2
NET "data_leds[3]" LOC = N14; // LD3
NET "data_leds[4]" LOC = P13; // LD4
NET "data_leds[5]" LOC = N12; // LD5
NET "data_leds[6]" LOC = P12; // LD6
NET "data_leds[7]" LOC = P11; // LD7

NET "dt_mag[7]" LOC = E6; // A2 pin 4 (Only pin used on bottom row)
NET "dt_sign[7]" LOC = D5; // A2 pin 5
NET "dt_mag[6]" LOC = D6; // A2 pin 7
NET "dt_sign[6]" LOC = D7; // A2 pin 9
NET "dt_mag[5]" LOC = D7; // A2 pin 11
NET "dt_sign[5]" LOC = D8; // A2 pin 13
NET "dt_mag[4]" LOC = D10; // A2 pin 15
NET "dt_sign[4]" LOC = B4; // A2 pin 17
NET "dt_mag[3]" LOC = B5; // A2 pin 19
NET "dt_sign[3]" LOC = B6; // A2 pin 21
NET "dt_mag[2]" LOC = A7; // A2 pin 23
NET "dt_sign[2]" LOC = A8; // A2 pin 25
NET "dt_mag[1]" LOC = B10; // A2 pin 27
NET "dt_sign[1]" LOC = B11; // A2 pin 29
NET "dt_mag[0]" LOC = A12; // A2 pin 31
NET "dt_sign[0]" LOC = A13; // A2 pin 33

NET "clk_16_Meg_in" LOC = D9; // A2 pin 35 (GCK4)
Appendix B

Python Code for Base Station

B.1 sig_acquire.py

```python
# -*- coding: utf-8 -*-
# INSTALL sudo aptitude install python-serial
import serial
import numpy as np
import datetime
import sys, os, errno
import math
import argparse

from tart.imaging import location
from tart.imaging import angle
from tart.operation import observation
from tart.operation import settings

# TODO Reduce latency between time stamp and opening of serial port
# so that the time stamp accurately reflects the start of the sampling process

# Emulate mkdir -p functionality in python
def mkdir_p(path):
    try:
        os.makedirs(path)
    except OSError as exc:  # Python >2.5
        if exc.errno == errno.EEXIST and os.path.isdir(path):
            pass
        else: raise

def create_timestamp_and_path(base_path):
```

```
# Time stamp information for naming files
ts = datetime.datetime.utcnow()

# Create a meaningful directory structure to organize recorded data
p = base_path + '/' + str(ts.year) + '/' + str(ts.month) + '/' + str(ts.day) + '/
mkdir_p(p)

# Call time stamp again (the directory name will not have changed, but the timestamp will be more
accurate)
ts = datetime.datetime.utcnow()
return ts, p

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Acquire data from the TART telescope.')
    parser.add_argument('--data-directory', required=True, help="The filesystem path for the telescope data.")
    parser.add_argument('--serial-port', default='/dev/ttyUSB0', help="The serial port for the data.")
    parser.add_argument('--read-size', type=int, default=2**17, help="The number of bits to record.")
    parser.add_argument('--config-file', default='telescope_config.json', help="The telescope configuration file.")
    parser.add_argument('--test', type=bool, default=False, help="Generate Fake Data.")

    args = parser.parse_args()
    base_path = args.data_directory
    read_size = args.read_size
    print "Reading %d data points" % read_size

    config = settings.Settings(args.config_file)

    if (args.test == False):
        ser = serial.Serial('/dev/ttyUSB0', 115200, timeout = 2)
        t_stmp, path = create_timestamp_and_path(base_path)

        # Send start signal to FPGA via RS232 to begin sampling
        # ascii character code for device control 1 (DC1) commonly used as XON
        ser.write(chr(17))

        # Read data from serial port
        data = ser.read(read_size)
        ser.close
        # Convert digits in string to integer array
        data = np.fromstring(data, dtype='uint8')
    else:

136
```
t_stmp, path = create_timestamp_and_path(base_path)
data = np.array(np.random.randint(0,255,size=2**16), np.uint8)
obs = observation.Observation(t_stmp, data, config)
filename = path + t_stmp.strftime('%H_%M_%S.%f') + '_data.pkl'
obs.save(filename)
```

B.1.1 observation.py

```python
import numpy as np
import datetime
import cPickle
import math

from ..imaging import tart_util
from ..imaging import location
import settings

# Antenna numbers are 1...8

class Observation:
    '''Antenna positions are going to be in meters from the array reference position. They will be in 3D ENU co-ordinates.'''
    def __init__(self, timestamp, data, config):
        self.timestamp = timestamp
        self.data = data
        self.config = config

    def save(self, filename):
        configdict = {
            'name': self.config.name,
            'array_orientation': self.config.array_orientation,
            'locations': self.config.locations,
            'num_antennas': self.config.num_antennas,
            'num_baselines': self.config.num_baselines,
            'geo': [self.config.get_lat(), self.config.get_lon(), self.config.get_alt()],
            'ant_positions': self.config.ant_positions,
            'frequency': self.config.frequency,
            'bandwidth': self.config.bandwidth,
        }
        d = {
            'config': configdict,
        }
```
d['timestamp'] = self.timestamp
d['data'] = self.data

save_data = open(filename, 'wb')
cPickle.dump(d, save_data, cPickle.HIGHEST_PROTOCOL)
save_data.close()

def get_antenna(self, ant_num):
    if (ant_num > self.config.num_antennas):
        raise "Antenna %d doesn't exist" % ant_num
    mask = (1 << (ant_num-1))
    ant_data = ((self.data & mask) / mask)
    ret_data = np.array(ant_data, dtype=np.int)
    ret = ret_data*2.0 - 1.0 # Convert to bipolar binary
    return ret

def get_sampling_rate(self):
    ref_freq = 16.368e6 # See the Max 2769 data sheet. We operate in one of the predefined modes
    return ref_freq

def get_julian_date(self):
    return tart_util.get_julian_date(self.timestamp)

def get_mjd(self):
    return tart_util.get_mjd(self.timestamp)

# Deprecated since we not want to store our own objects (that might change)
# def Observation_Load(filename):
#    load_data = open(filename, 'rb')
#    ret = cPickle.load(load_data)
#    load_data.close()
#    return ret

def Observation_Load(filename):
    load_data = open(filename, 'rb')
    d = cPickle.load(load_data)
    load_data.close()
    ret = Observation(d['timestamp'], d['data'], settings.from_dict(d['config']))
    return ret

import unittest

class TestObservation(unittest.TestCase):
    def setUp(self):
        # Simulated FPGA serial data
self.data = np.array(np.random.randint(0,255,size=2**16), np.uint8)

ts = datetime.datetime.utcnow()

self.config = settings.Settings('../../tools/operation/telescope_config.json')

self.obs = Observation(ts, self.data, self.config)

def test_load_save(self):
    self.obs.save('data.txt')

    # Array to put the data into
    antenna_data = np.zeros((len(self.data), self.config.num_antennas), dtype=np.int8)

    for i in range(self.config.num_antennas):
        antenna_data[:, i] = self.obs.get_antenna(i + 1)

    nobs = Observation_Load('data.txt')
    self.assertTrue((self.data == nobs.data).all())
    self.assertTrue((self.obs.get_antenna(1) == nobs.get_antenna(1)).all())
    self.assertEqual(self.obs.get_julian_date(), nobs.get_julian_date())

B.1.1.1 telescope_config.json

```
{
    "name": "TART_Site::Physics_Rooft",
    "frequency": 1.57542e9,
    "bandwidth": 2.0e6,
    "lat": -45.86391200,
    "lon": 170.51348452,
    "alt": 63.0,
    "num_antenna": 5,
    "orientation": -22.0,
    "axes": ["East", "North", "Up"],
    "locations": [
        [0.572, -0.415, 0.0],
        [0.0, 0.0, 0.0],
        [-0.302, -0.233, 0.0],
        [0.287, 0.392, 0.0],
        [-0.64, 0.605, 0.0]
    ]
}
```