Circuits and Systems
Mekelweg 4,
2628 CD Delft
The Netherlands
http://ens.ewi.tudelft.nl/

CAS-2011-006

M.Sc. Thesis

Fundamentals, Specifications, Architecture and Hardware Towards a Navigation System Based on Radio Pulsars.

Vashishth Krishan Chaudhri

Abstract

Pulsar based navigation is a novel area of research, and might once offer the possibility to man of safely traveling distances much beyond Earth. Recently, there has been work on a receiver tracking algorithm that makes use of X-ray pulsar signals [1]. However, a system based on this would not find use on celestial bodies where an atmosphere blocks the harmful X-rays.

This thesis is directed towards building a navigation system in hardware that makes use of 'radio pulsars'. The navigation system may allow a receiver observing radio pulsar signals to determine its position with respect to a reference, possibly with the receiver being anywhere in the universe of interest. The system is initially intended for estimating a receiver's coordinates without a prior knowledge of the receiver position, pulsar locations relative to the receiver, as well as the expected pulsar signals. This thesis lays down the feasibility and the limitations of the system that would use radio pulsars. The work presented includes: defining the fundamentals of radio pulsar based navigation, selecting suitable signal processing algorithms, analyzing system usage scenarios and related SNR enhancement by subsystems, constructing the system architecture, and discussing a hardware prototype based on FPGAs. Another objective (which has been successfully completed) is to rate the methodology of automatic RTL generation from Simulink models.

Faculty of Electrical Engineering, Mathematics and Computer Science **Delft University of Technology**

Fundamentals, Specifications, Architecture and Hardware Towards a Navigation System Based on Radio Pulsars.

Thesis

submitted in partial fulfillment of the
requirements for the degree of

Master of Science

in

Microelectronics

by

Vashishth Krishan Chaudhri
born in New Delhi, India

This work was performed in:

Circuits and Systems Group
Department of Microelectronics & Computer Engineering
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Delft University of Technology
Department of
Microelectronics & Computer Engineering

The undersigned hereby certify that they have read and recommend to the Faculty of Electrical Engineering, Mathematics and Computer Science for acceptance a thesis entitled **"Fundamentals, Specifications, Architecture and Hardware Towards a Navigation System Based on Radio Pulsars."** by **Vashishth Krishan Chaudhri** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: 30 September 2011

Chairman and Committee Member: ____________________
prof.dr.ir. A.J. van der Veen

Advisor and Committee Member: ____________________
dr.ir. N.P. van der Meijs

Committee Members: ____________________
dr.ir. C.J.M Verhoeven

____
dr.ir. T.G.R van Leuken

____
ir. P.J Buist

____
ir. S. Engelen

Abstract

Pulsar based navigation is a novel area of research, and might once offer the possibility to man of safely traveling distances much beyond Earth. Recently, there has been work on a receiver tracking algorithm that makes use of X-ray pulsar signals [1]. However, a system based on this would not find use on celestial bodies where an atmosphere blocks the harmful X-rays.

This thesis is directed towards building a navigation system in hardware that makes use of 'radio pulsars'. The navigation system may allow a receiver observing radio pulsar signals to determine its position with respect to a reference, possibly with the receiver being anywhere in the universe of interest. The system is initially intended for estimating a receiver's coordinates without a prior knowledge of the receiver position, pulsar locations relative to the receiver, as well as the expected pulsar signals. This thesis lays down the feasibility and the limitations of the system that would use radio pulsars. The work presented includes: defining the fundamentals of radio pulsar based navigation, selecting suitable signal processing algorithms, analyzing system usage scenarios and related SNR enhancement by subsystems, constructing the system architecture, and discussing a hardware prototype based on FPGAs. Another objective (which has been successfully completed) is to rate the methodology of automatic RTL generation from Simulink models.

Acknowledgements

I take this opportunity to thank several people with whom I have had the privilege of being associated with during my MSc. thesis project.

My sincere thanks to my teacher and guide, dr.ir. N.P. van der Meijs. I have been extremely fortunate to have been under his tutelage. I met him to discuss possible topics for my thesis. He welcomed me to do a thesis under him without giving it a second thought. Throughout my thesis, he frequently monitored, motivated, and at times pushed me to do better than I could. This thesis has been possible because of his great teaching and training me to think, write and judge more scientifically.

Many thanks to ir. Steven Engelen and ir. Arash Noorozi for their beneficial support in helping me understand the prior work related to this thesis. They always took out as much time I needed to discuss and solve questions. Thank you for your patience and motivation.

Special thanks to ing. Antoon Frehe for his invaluable assistance in providing me with excellent computers and software to work with. I was amazed many a time with his brilliance of Linux, and he was my model of a great engineer during my study.

I thank my seniors ir. Chockalingam Veerappan and ir. Sumeet Kumar for helping me get accustomed to the work environment of the CAS group. It was always nice to be in their company for their excellent suggestions and guidance.

Had it not been for Xianli Ren, I would have probably had a less enjoyable time in the Lab. I had an enriching period of stay with him, especially during our innumerous discussions on all sorts of topics known to us. We helped each other solve all issues, and I hope our friendship would last forever.

Towards the end, I thank my parents and sister with all my heart. It was only due to their perseverance that I enrolled for an MSc. education. Irrespective of my shortcomings, they have always supported and stood by me completely. They have sacrificed many things allowing me to see this beautiful day. I dedicate the better parts of this work to them.

Vashishth Krishan Chaudhri
Delft, The Netherlands
30 September 2011

Contents

List of Figures

List of Tables

List of Abbreviations

A to D	Analog to Digital
ASIC	Application Specific Integrated Circuit
BPFB	Band Pass Filter Bank
CPU	Central Processing Unit
DDR	Double Data Rate
DM	Dispersion Measure
DSN	Deep Space Network
DSP	Digital Signal Processing
EDA	Electronic Design Automation
FFA	Fast Folding Algorithm
FFT	Fast Fourier Transform
FPGA	Field Programmable Gate Array
GPS	Global Positioning System
GPU	Graphics Processing Unit
GUPPI	Green Bank Ultimate Pulsar Processing Instrument
HDL	Hardware Description Language
HLS	High Level Synthesis
IFFT	Inverse Fast Fourier Transform
IP	Intellectual Property
ISE	Integrated Software Environment
ISM	Inter Stellar Medium
JPL	Jet Propulsion Laboratory
LOFAR	Low Frequency Array for Radio Astronomy
NASA	National Aeronautics and Space Administration
OEM	Original Equipment Manufacturer
PFB	Polyphase Filter Bank
RAM	Random Access Memory
RTL	Register Transfer Level
SNR	Signal to Noise Ratio
TOA	Time of Arrival
VHDL	Very High Speed Integrated Circuits Hardware Description Language
XNAV	Autonomous X-ray Pulsar based Spacecraft Navigation

Introduction 1

1.1 Introduction to the Thesis

This work is towards the dream of realizing a navigation system in hardware that operates based on radio pulsars. The principle of such a navigation system is similar to that of GPS based navigation wherein remote satellites orbiting Earth provide information allowing a receiver to determine its position. In radio pulsar based navigation, we aim to use the information transmitted by radio pulsars to estimate the coordinates of a body in space. The primary principle of navigation using pulsars is based upon determining the Time of Arrival (TOA) of three separate signals with known individual signal sources [1][2]. Observing the signals from radio sources in different directions will enable a receiver with an accurate clock (for the reference of time) to estimate its positional coordinates with respect to an inertial reference system.

1.2 Motivation

The motivation to devise such a system is manifold. Ideally, a pulsar based navigation system should be of use when the observation of pulsar signal sources is possible. Alternatives to current GPS based navigation on Earth are being sought to improve navigation capabilities in terms of reliability. The system could allow satellite free navigation on Earth for sea ships.

Also, as is the belief that in future man and his experiments would travel to distances much beyond the realms of Earth, a universal navigation system becomes an essential requirement for such missions. One can think of navigation on planets apart from Earth as well as other celestial bodies by making use of such a system. Overall, when one considers navigation in space beyond the realms of our Solar System, such a navigation system would prove to be of considerable advantage.

1.3 The Navigation System Challenges

Although seemingly simple in principle, there are several hurdles that are needed to be overcome in realizing such a navigation system. The main challenges when using radio pulsars for navigation are the following:

1. The extremely weak pulsar signal strength that is being used to navigate. The received signal to noise ratio on Earth is very small indeed. This is because the pulsar signals are emitted many light years away from Earth, leading to addition of noise and distortion due to the propagation channel as will be discussed in Section 2.5.

2. Another major challenge arises due to an unknown receiver velocity with respect to the pulsar. As is implied by the term navigation, we must be able to locate our coordinates in time when the pulsar signal is visible. Complications due to Doppler shift [3] arise for the case of an unknown velocity of the receiver when estimating coordinates. This leads to the resulting observed information of the pulsar signal to be an unknown different from that which is observed with respect to a known inertial frame of reference. Thereby, the Doppler shift demands techniques that compensate for the effect. To detect the unknown velocity, one requires a lot of computation power in hardware, as is analyzed in this thesis.

1.4 Thesis Contributions

Initial work towards the navigation system had been undertaken in [4] and [2]. Their proposals classify the required digital hardware system into two separate 'ends'. The front end should be able to detect a pulsar and determine the velocity at which the receiver is moving relative to the pulsars. The back end should compute the position of the receiver while maintaining communication with the front end. The communication between the two will be explained in Section 2.6.

This thesis is primarily towards devising the fundamentals, specifications and hardware of the front end. The thesis has a side objective of evaluating Simulink based generic RTL generation; serving as an aid to determine the most efficient methodology for designing the system hardware.The contributions of the thesis include:

1. Reviewing the current state of the art pulsar astronomy techniques.

2. Selecting the algorithms that are suitable for use in our navigation system.

3. Constructing the system architecture.

4. Defining implementation scenarios and SNR specifications for subsystems.

5. Evaluating the Simulink based RTL generation methodology.

6. Determining suitable hardware for a practical implementation.

1.5 Thesis Outline

Apart from this introduction, the thesis consists of six chapters. Chapter 2 presents the theoretical background as well as an overview of the current technology related to radio pulsar based navigation. Some topics discussed include: pulsar signals, current deep space navigation methods, propagation channel and effects, initially proposed system, and state of the art radio astronomy instruments.

Chapter 3 discusses the different design choices towards a real world implementation of the signal processing front end. The chapter also puts forth the modular architecture as should be implemented in hardware.

Chapter 4 derives the SNR improvement of the pulsar signal by different signal processing steps. This is done considering specific system usage scenarios. This chapter helps put into perspective the possibility of conquering the poor pulsar SNR allowing a successful system realization.

Chapter 5 calculates the computational requirements of a subsystem for different navigation scenarios, proposes its FPGA targeted architecture, and estimates the needed hardware to implement the same.

Chapter 6 assesses the benefit of using automatic RTL generation methodology from Simulink models. A choice is made amongst different available tools for this purpose. The Simulink HDL Coder is chosen, and its performance and features are analyzed from a hardware implementation perspective. This analysis in part has been performed by constructing a required signal processing block using the tool.

Chapter 7 describes the future work that needs to be done in order to realize the system. Also, a summary based on this thesis has been presented.

Theoretical Concepts and Current Technology 2

2.1 Pulsars

2.1.1 Pulsars and the signals they emit

Pulsars are rotating neutron stars that emit electromagnetic radiations. These stars are located at several different locations in our Galaxy. Even though studies of pulsars thus far has allowed man to observe the properties of the emitted signals, the understanding of the pulsar structure as well as the signal emission mechanism remain unanswered questions [5].

A simple model that allows one to account for some of the observed signal properties is the cone shaped emission beam structure centered on the magnetic axis of the pulsar. A pulsar signal on Earth is observed to contain peak like structures. A pulsar rotates in a periodic manner, which results in such a signal whenever the conical emission beam crosses the Earth. This is similar to the working of a lighthouse where a beam of light crosses an observer after a specific time interval. In congruence to the light house effect, the received pulsar signal displays a constant period. A pictographic representation of the model can be seen in Figure 2.1.

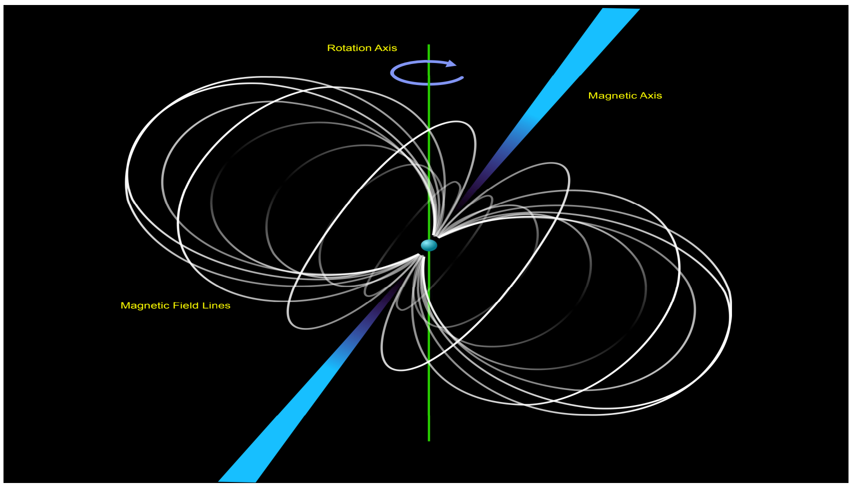


Figure 2.1: Pulsar signal emission

Although surprising but true, a pulsar signal's period is highly stable and is compared to those of atomic clocks on Earth [6]. The real world Vela pulsar's signal can be seen in Figure 2.2 [7]. This signal was observed using the Green Bank telescope in September, 1970. One can clearly observe the period of the signal, T_0.

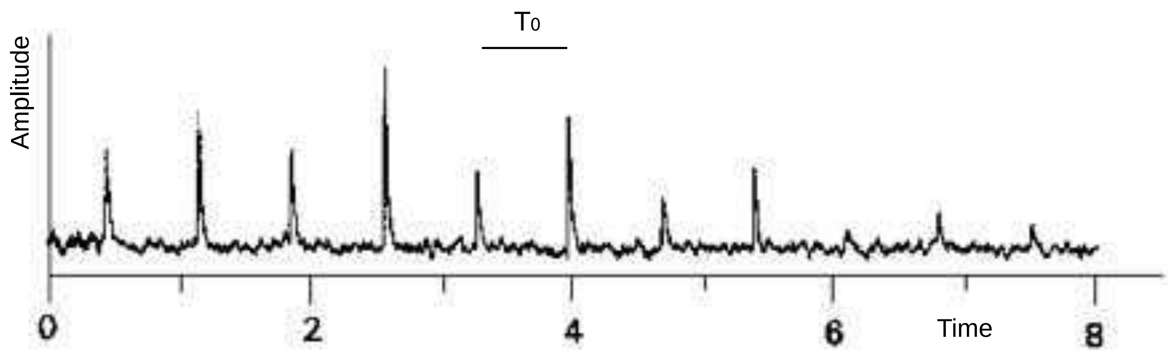


Figure 2.2: The Vela pulsar signal based on [7]

A pulsar emits a signal that is spread over a very large frequency band. Based on the range of the frequencies that are emitted, a pulsar can be classified, amongst others, as an X ray signal source or a radio signal source. Initially, navigation using both radio pulsars and X-ray pulsars had been studied. Since the 1980's, work has largely been carried out only towards using the X-ray pulsars [8]. This has been primarily because X-ray pulsars emit signals with a better signal to noise ratio (SNR) than radio pulsars and the conceived purpose of such a navigation system was only for deep space navigation.

The research of this thesis is targeted towards devising a navigation system based on radio pulsars. Such a system has useful possibilities that are not offered by the X-ray based system. A navigation system based on radio pulsars could find use in vehicle navigation on Earth as well as rover tracking on other planets where X-rays are blocked by the atmosphere.

2.1.2 Pulsar signal dimensions

Radio pulsars observed on Earth have been found to display periods that range from 1.4 milli-seconds to 8 seconds [9][10]. For use as a signal source in our navigation system, one should prefer milli-second pulsars. This will be clear from Sections 4.2.1 and 4.2.2. New pulsars are being discovered continuously and hence we have an ever increasing milli-second pulsar database.

The width of the emitted signal is generally a small fraction of the total period. A typical width of the signal would be around 5 percent of the total period. However, due to interstellar propagation, see Section 2.3, the width that is input to the navigation system can be considerably bigger.

The amplitude of the received signal is defined relative to the noise, and is a random parameter. The signal to noise ratio depends on the observed frequency as well. This will be presented in context to our navigation system in Section 3.4.

2.1.3 Period glitches

Although the period of a pulsar signal is very stable, a pulsar can occasionally undergo a phenomenon called 'glitch'. This phenomenon is observed usually in younger pulsars as compared to the more stable older pulsars. A glitch relates to a sudden random increase in the pulsar period [5]. Subsequently however, the pulsar period slows down exponentially to its original period before the occurrence of the glitch.

2.1.4 Period decays

A pulsar period although stable, is not perfectly immune to decay. All pulsars have a characteristic ‘spin-down’ rate. This spin down rate is extremely small and is of the order of roughly 10^{-13} to 10^{-19} seconds per second [4]. The primary reason for spin-down is due to the loss of energy of the neutron star.

2.2 The Pulsar Frame of Reference

The location and signals of pulsars are recorded with respect to the solar system barycenter at the J2000 epoch [2]. Since the ultimate goal of devising the navigation system is to find the coordinates of a body, we must first choose a reference system for the same. The pulsar data that we would be using will either be that recorded with respect to the solar system barycenter, or it would be based on a new reference system. Our navigation results will hence form an automatic link with the used reference system.

2.3 Current Deep Space Navigation Methods

2.3.1 Deep space network

The Deep Space Network (DSN) is an Earth based navigation system for probes that are sent into the Solar System. The network consists of three communication systems that are located approximately 120 degrees apart on the Earth’s surface. The antennas are currently located in: Mojava desert of California, Madrid of Spain, and Canberra of Australia. This network is responsible for tracking deep space missions such as the Phoenix Mars Lander [11], and is the current dominant method of navigation for space probes [12].

2.3.2 Autonomous X-ray pulsar based spacecraft navigation (XNAV)

The idea of navigation using X-ray pulsars has been known since the time NASA’s Jet Propulsion Laboratory (JPL) conducted research towards this in the 1970’s and 80’s [13]. This method of navigation is an area of active research, although it is predominantly being contributed to by a few organizations. The motivations for such a navigation system have been highlighted for use in missions beyond Jupiter where the DSN accuracy degrades, missions that may require real time on-board navigation, and lastly to reduce burden on the DSN.

The system relies on input X-ray particles. The pulsar signal is observed when the input count rate of particles becomes larger as compared to the average count rate. As has been mentioned before, X rays are blocked by the atmosphere, and hence such a navigation system would find use only in space above the Earth. Also, a signal’s phase content is not as precisely obtained as compared to radio pulsar based navigation which could cause an increase in error during the navigation process [14]. The important work that has been undertaken in XNAV [1] is more of tracking a receiver using X-ray pulsars. It assumes a knowledge of locations of X-ray pulsars relative to the receiver as well as their initial observed characteristics. This is instead of conducting a live navigation without the knowledge of the initial receiver position, which has been kept as a target for our system.

2.4 The SNR Challenge for Pulsar Based Navigation

As is mentioned in the introduction, pulsar based navigation faces two challenges namely; a very poor input SNR to the system and requirements of very large computational resources. The SNR of the pulsar is extremely poor, making it difficult to even detect it for navigation purposes. For example, at an observing frequency of 1 GHz and an effective receiving area of 10 m^2 using a directed antenna, according to [15] amongst all pulsar candidates, we have at best an SNR that is approximately equal to -50 dB in magnitude.

A natural question that now arises is how are radio pulsars observed in science in the present day? In the current astronomical observations, a radio pulsar signal is detected using large telescopes and complex signal processing. A large telescope ensures an improved input SNR to the later signal processing engine. Since radio observatories correct for their velocity with respect to the pulsar (a term known as Barycentering) [16][17], the number of computations required as compared to our case of navigation using pulsar signals are also reduced by a large factor.

This challenge needs to be taken into consideration when making design choices. We need to adhere to algorithms that offer a better enhancement of the pulsar SNR instead of their counterparts.

2.5 Signal Propagation Channel and Dispersion

A pulsar signal travels very large distances on its way to reaching our planet. Pulsars are located at several hundred or in other cases, several thousand light years away from Earth. A signal passes through the intergalactic space, which is scientifically known as the Interstellar Medium (ISM). A pulsar signal that is observed on Earth is found to have been under the influence of different effects leading to: Dispersion, Scintillation, and Scattering. These effects are discussed in the following text.

2.5.1 Scintillation

Scintillation is the term given to the observation of random signal strengths of the received pulsar signal in time. The effect is similar to that present when observing stars in the optical domain. One encounters a ‘twinkling’ of stars in the night sky when observed with the naked eye, which is again due to a variation in the strength of the signal received.

The strength of the received signal is a measure of its amplitude. Since the amplitude received is random, the SNR that is commonly specified for a pulsar should be an average received value over multiple observations.

2.5.2 Scattering

The term 'Scattering' is commonly used in Telecommunications to describe the propagation of the signal in question along different paths before being received. Similarly, a pulsar signal also travels different path lengths, which leads to there being a difference in the arrival times of individual rays [5]. Such an effect causes broadening of a natural sharp pulse. Previous research has shown the presence of a strong correlation between Scattering and 'Dispersion Measure' (DM), which in turn is discussed in the next section.

2.5.3 Dispersion

Dispersion is a phenomenon in which spectrum components of the pulsar signal undergo a frequency dependent delay along the path length traveled. The ISM acts as a 'phase only filter' with a specific transfer function, say H. In all cases, the delay in the pulse arrival times of the signal is dependent on the observing frequency. A parameter relating the two known as the Dispersion Measure, is a product between the average electron density along the line of sight to the pulsar and the path length traveled by the signal. The interstellar transfer function is elaborated when explaining the technique of coherent dedispersion in Section 2.6. Figure 2.3 represents the concept of a frequency dependent pulse arrival time for a narrow band of the signal. A real world dispersed pulse profile can be seen in [18].

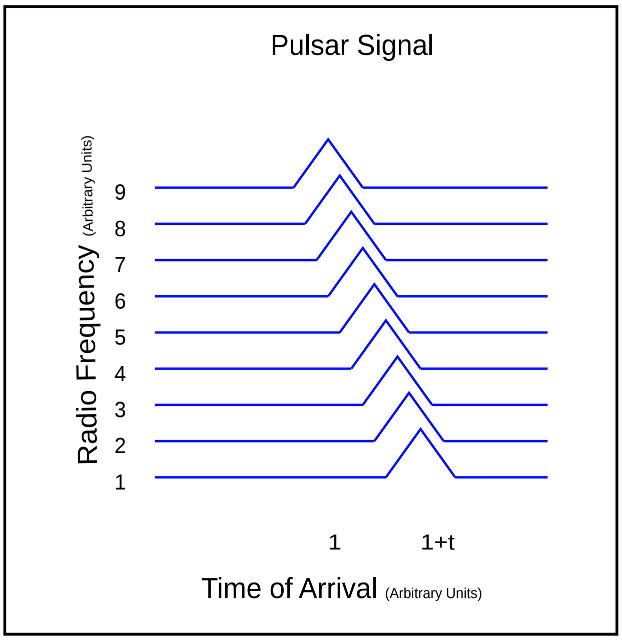

Pulsar Signal
Radio Frequency (Arbitrary Units)
9
8
7
6
5
4
3
2
1
1
1+t
Time of Arrival (Arbitrary Units)

Figure 2.3: Dispersion: frequency dependent delay

Amongst these effects, the system proposed in [4] corrects for the dispersion caused by the ISM when detecting a pulsar. See Sections 2.6 and 2.7 for more details.

2.6 Initial Navigation System as Formerly Proposed

2.6.1 System

The work in [4] and [2] propose some processing steps for navigation using radio pulsars. The system proposed uses the Doppler principle to undertake navigation. The velocity related changes in the signal period due to Doppler are proposed to determine the position of the receiver.

The work in [4] recommends the possible use of three processing steps namely; Dedispersion, Folding, and Correlation. These steps are to determine the period of a specific pulsar as observed at the receiver. Likewise, [2] proposes the use of the Unscented Kalman Filter and a Runge-Kutta-Nystrom Integrator to accurately determine the position of the receiver.

The proposed navigation system can be understood with the help of the block diagram presented in Figure 2.4. What follows is a discussion of the working principles of the proposed steps with the help of other standard radio astronomy literature.

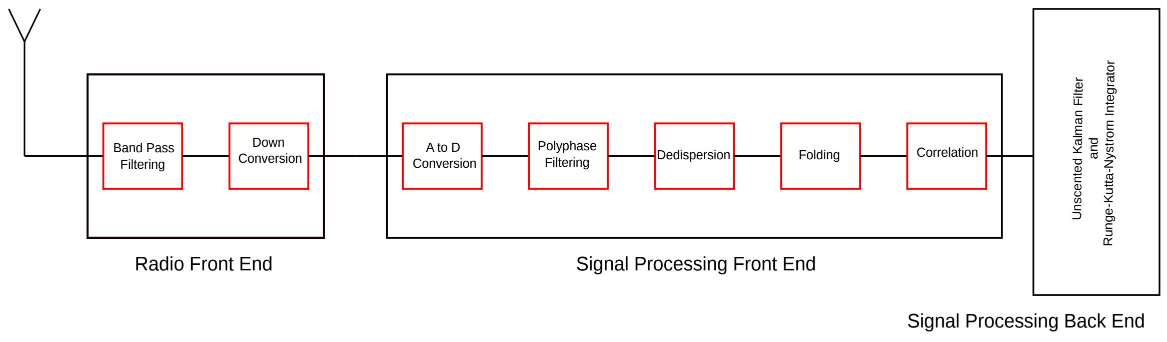


Figure 2.4: System block diagram

The observation of a pulsar signal starts with the help of a receiving antenna. After this step, one would implement radio frequency domain procedures such as band pass filtering and down conversion of the input frequency band [5]. The signal will then be sampled and quantized using an A to D converter. The sampling process should be performed taking the Nyquist sampling criterion into consideration. Following these would be signal processing steps such as polyphase filtering [19] (to split the input broadband signal into frequency sub-bands) and 'dedispersion' [20][5].

De-dispersion is a procedure to correct for the effects due to the interstellar channel. The method involves correction for the channel effects using either of the techniques: Incoherent or Coherent dedispersion which are discussed in Section 2.7. Once the pulsar signal has been de-dispersed, we would have as proposed in [4], the 'Folding algorithm' [21] implementation. The significance of folding is to generate a pulsar profile which can be compared to known pulsar profiles by the process of correlation. This is explained in the subsequent paragraphs.

In the process of folding, a real time de-dispersed pulsar signal acts as an input to the folding subsystem. The input would comprise of a discrete time series of data. The sampling frequency of these discrete samples would be either equal to that output by the A to D converter, or be at a user specified lower scaled rate. Since the pulsar signal displays a constant period in real time, we would receive the input one period after the other. As the period is very stable [6], the phase of the pulsar signal would be the same in subsequent periods. The inherent principle of folding is that the received signal corresponding to a single period of the radio wave is added to the already accumulated signal periods (in an accumulator). In this process, it is imperative to realize that we are in time adding the pulsar signal values to the previously received pulsar signals. Thereby, one keeps on adding pulsar signals and attains a signal that has a larger amplitude. Therefore, we obtain a better SNR [21].

Since pulsars are emitters with highly stable periods [6], there would be a deviation in the observed period from the originally emitted one because of a receiver velocity in accordance to the 'Doppler effect' [3]. Comprehensively stating, for a given pulsar period and pulse profile, the signal processing front end would need to detect for several different profiles each unique to a particular, velocity dependent, Doppler period. The different periods that need to be searched for would be chosen based on the possible velocities with which the body would be traveling with respect to the pulsar. The resulting pulse shapes after dedispersion and folding would be then correlated with known pulsar templates that would depict the shapes as seen at different velocities. The pulsar template corresponding to the maximally correlated value would be then chosen as a representative of the velocity at which the body is moving. This would enable the 'signal processing back end' to identify the pulsars which we are tracking and allow it to accurately determine the coordinates of the body.

A detailed discussion of the proposed signal processing back end procedures is beyond the scope of this document. However, its responsibilities and communication with the front end are discussed below.

In [4] and [2], the communication between the front and back end pertains to an exchange of namely four types of data. These are: the pulsar templates, the pulsar periods to fold at, the velocities at which the receiver maybe moving, and the observed period with respect to the pulsar. Of these, the first three are communicated by the back to the front end and the last data type is messaged vice versa as seen in Figure 2.5.

The system corrects for the Doppler effect by performing dedispersion, taking into consideration the velocity at which the body maybe moving. This is performed in parallel for different possible pulsars that are to be detected by the back end, each at several different velocities. The set of velocities that are needed to be taken into account are provided by the back end and is based on the current velocity estimates of the body. In other words, the back end transmits velocities which are those at which the receiver maybe be moving with respect to the pulsar. These velocities are determined by the back end. Once the signal has been dedispersed, a folding is said to be performed at the Doppler period which is expected to be observed for the corresponding velocity messaged by the back end. Again, the folding would also be performed for different pulsars, just after the dedispersion process step. This is confusing since one

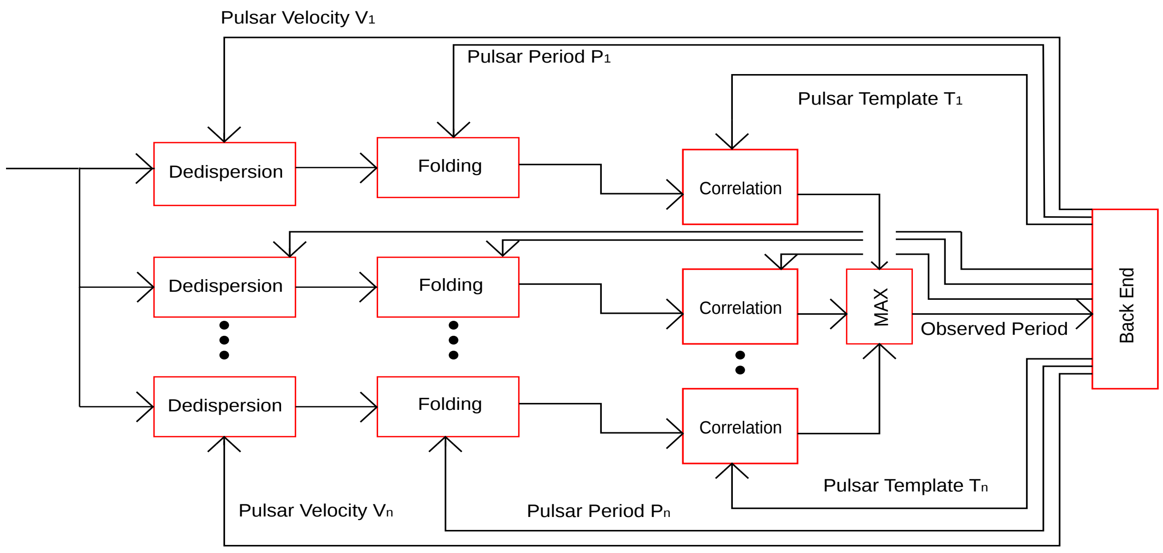


Figure 2.5: Front-Back end communication

has already corrected for the Doppler effect during dedispersion. Hence a change to the communication architecture is proposed in Section 3.2.1.

Following the generation of a stable pulse profile on folding, the profile is correlated by a template. The back end has in store several different pulsar templates. For each real world pulsar signal, the back end has templates of the signal as would be seen at different velocities of interest. Only those templates of a pulsar are messaged to the front end that correspond to the velocities being taken into account at a given instant of time.

For each pulsar, the template that yields the highest correlation is chosen to depict the velocity at which the body is moving with respect to the pulsar. For each pulsar that is being searched for, the Doppler period corresponding to the highest correlated velocity is messaged to the back end. The back end subsequently uses this information to update the coordinates of the receiver.

2.6.2 Possible usage limitation

Based on the work in [4], it was thought that it might be difficult to navigate without an initial knowledge of the pulsar locations relative to the receiver. Indeed, as seen in Section 5.2.1, the computational resources needed for such cases is too large for an efficient implementation. As presented in Section 5.2.2, we then restrict ourselves to the scenario where we know the initial coordinates of the receiver and the initial velocity direction relative to the host planet at the start of navigation; allowing us to reduce the number of computations we must otherwise perform.

2.7 Dedispersion Techniques

As is mentioned in the previous section, there are two different dedispersion techniques. The difference between the two will become evident once both are discussed. The first technique: Incoherent dedispersion, is discussed below.

2.7.1 Incoherent dedispersion

The concept of this technique is simple. As is mentioned before, the constituent frequencies of a pulsar signal arrive at the receiver at different times. In this technique, the input frequency spectrum is split into smaller individual frequency bands. An entire band is then delayed by a specific time which corrects for the delay due to dispersion relative to the maximally dispersed band. The bands are then added to each other, resulting in a dedispersed pulse profile.

The difference in the time of arrival between two frequencies (f_1 and f_2) is presented in [5] as

$$\Delta t \approx 4.15 * 10^6 ms * (f_1^{-2} - f_2^{-2}) * DM \; . \tag{2.1}$$

Based on this formula, one can calculate the required delay for a particular band relative to the band arriving last. Since we delay all the frequencies comprising a band by the same margin, one should choose the center frequency of the band for calculations. This makes it clear that the spectrum is never precisely dedispersed. The advantage of this procedure is however, the low complexity of implementation.

2.7.2 Coherent dedispersion

This technique accounts for the behaviour of the ISM as a filter. The method involves correction for the channel effects by passing the input frequency band through the inverse of the interstellar transfer function. A derivation of the interstellar transfer function ('H') is beyond the scope of this document. For our knowledge, the mathematical formula is presented below [5]

$$H(f_0 + f) = e^{i\frac{2D\pi}{(f_0+f)f_0^2}DMf^2} \; . \tag{2.2}$$

Here, f_0 is a frequency relative to which the dispersion at another frequency f is defined. The Dispersion Measure DM has been explained before. D is the dispersion constant, and is a dependent on the interstellar plasma frequency, the speed of light, as well as the average electron column density along the path traveled [5].

As a proof of concept, the approach of coherent dedispersion has been evaluated using a real world pulsar. The data of the pulsar B0329+54 observed using the LOFAR telescope, has been used to perform an inverse filtering in software. Even though the used data contained a very small bandwidth of frequencies, nearing .5 MHz, one can observe the generation of a pulse profile upon dedispersion and folding. The result shown in Figure 2.6 is as seen after four hours of dedispersion and folding the pulsar data using Matlab. The simulation time is less than one minute on a 32 bit Windows machine. Compared to the experiment in [22], we have reduced the pulsar detection time by making use of a transfer function defined for the complete bandwidth as well as by the use of the overlap-save discrete convolution technique [5][17].

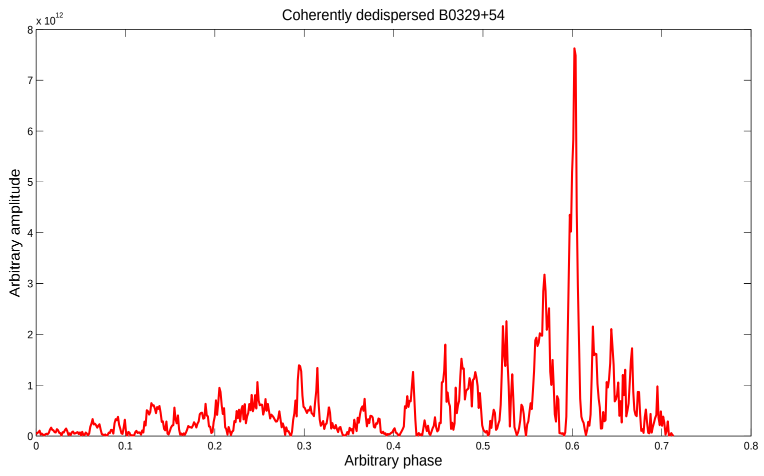


Figure 2.6: Inverse filtering of B0329+54 in software

Numerically, the inverse filtering operation can be performed both in time as well as the frequency domain. It is advantageous to implement dedispersion in the frequency domain. This is because, the computational complexity is reduced as compared to a time based implementation, which is desirable when processing wide bandwidths. In radio observatories around the world, dedispersion is regularly implemented. All coherent dedispersion procedures take place in the frequency domain. An introduction to the current technologies of radio astronomies is presented in the following section.

2.8 Radio Observatories Technology

Radio astronomy includes the science of observing radio pulsars. The goal of such studies is to find new pulsars, understand their behaviour, and answer related questions of physics. There are several observatories today that have enormous sized antennas, allowing the detection and accurate characterization of pulsars. The digital signal processing steps that are adopted to detect a pulsar involve filtering, dedispersion, and folding.

Current state of the art dedispersion instruments are commonly developed using a cluster of FPGAs. An example is GUPPI [23] that processes upto 800 MHz of signal bandwidth. The computational power required to dedisperse this bandwidth is approximately .5 TFLOPS [24]. The data once dedispersed is stored in memory. The subsequent folding operation is then performed offline using a cluster of CPUs or GPUs.

2.9 Conclusion

Pulsar signals exhibit a highly stable period. A pulsar signal is acted upon by the ISM causing unwanted effects such as dispersion. When using antennas of size nearing 10 m^2, the received radio pulsar signals have on average an SNR much smaller than -50 dB. This causes radio pulsar based navigation to be a challenging field of research. The possible navigation system processing steps, system architecture, and other system parameters are needed to be researched on in detail; entailing their use for navigation.

Design Choices and Proposed Architecture 3

3.1 Design Choices

3.1.1 Which frequencies to use? The lower or the higher?

Pulsars are broadband emitters. The range of frequency emissions that have been observed lie roughly from 100 MHz to 100 GHz [5]. In radio astronomy, it is very common to search for pulsars at frequencies nearing 400 MHz as well as 1400 MHz. Since we can only process a small band out of the entire spectrum, we must decide which range of frequencies to choose from for our navigation system.

Pulsar emissions at lower frequency ranges are broader at the receiving antenna than those at higher frequencies due to dispersion. For the case when we are folding the pulsar signal for a very long time during navigation, a broader pulse is actually desired since it would allow us a better improvement in SNR. See chapter 4 for more details. The pulse width however cannot be the only factor determining the suitability of the frequency band.

One can note from [15] and [5] that although the flux density of signals is inversely proportional to the frequency following a power law, the obtained SNR also depends on the system temperature and the effective antenna area. For many pulsars, the SNR observed is seen to be better at lower frequencies than at higher frequencies. In some cases however, it is advantageous to detect pulsars at higher frequencies. This is because pulsar signals originating near the Galactic plane suffer from large background temperatures [25] at lower frequencies, which leads to a reduction in the SNR. It is better to observe such pulsars at a higher frequency. Also, if one prefers to use the technique of incoherent dedispersion instead of coherent dedispersion, a higher frequency band can be more advantageous. This is because, higher frequencies suffer lesser dispersion.

Hence, the frequency band that we should use would depend on the location of suitable pulsars relative to the Galactic plane as well as the adopted dedispersion procedure. The next section presents the down conversion of the pulsar signal and the observed signal polarizations.

3.1.2 Down conversion

Irrespective of the frequency of observation, the input spectrum needs to be converted to baseband. This would allow further signal processing at reduced sampling speeds. The down conversion is in general performed by mixing the spectrum with a local oscillator having an oscillating frequency equal to the center frequency of the spectrum. This results in the generation of real and imaginary components of the signal, which are then passed through the inverse filter. There can also be other local oscillator frequencies that result in either a completely real, or a completely imaginary signal. However, the inverse filtering process is not affected by the down conversion procedure [5].

3.1.3 Digital or analog filter bank?

In the system block diagram, Section 2.6, the polyphase filter is placed after the A to D converter signaling that it takes place in the digital domain. There is however, an alternative in the form of an analog filter. The primary advantage of an analog filter is that it would reduce the computational requirements of the system as well as increase the speed of operation. However, one has to take into consideration that the filter needs to be implemented on a hardware platform. When using an FPGA based system, the digital filter offers the possibility of reconfiguration and easy integration. Moreover, digital filters have a flatter response in the pass band and also have better roll-off and stop band characteristics. For the same reasons, the choice of filtering is set to be in the digital domain.

3.1.4 An ideal model of the system ADC

In the overall navigation system architecture, an analog to digital converter will be an interface between the frequency down converter and the subsequent digital filter. As is commonly known, conversion from the analog to digital domain leads to a reduction in the SNR of the input data signal due to quantization and other effects. In our case, the data signal of the pulsar is embedded in a lot of noise as can be seen in Figure 3.1.

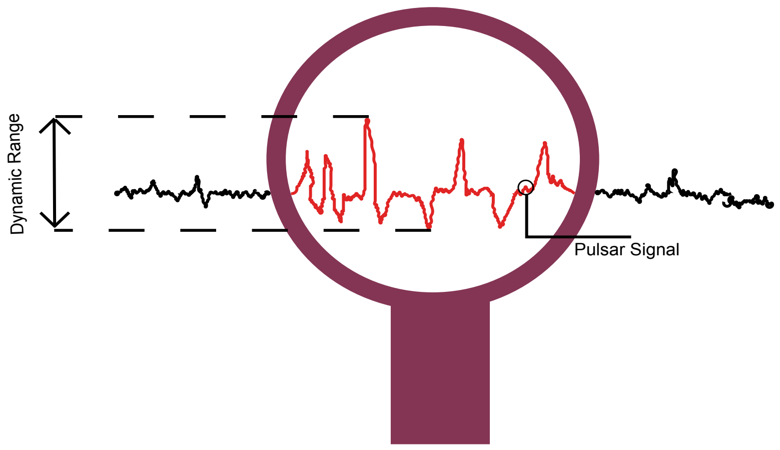


Figure 3.1: The noisy signal

A quick calculation needs to be performed to determine if current state of the art A to D converters have specifications that would be sufficient for our application. Our pulsar signals would have an SNR better than or near to -70 dB after being received by an antenna of size 10 m^2. This is because we require approximately 50 pulsars to be observed at any time instant, see Section 5.2.1. In appendix A, we can see that there are several pulsars with an SNR within this range. Since new pulsars are being discovered, in future we would probably have at least 50 such pulsars that would be uniformly distributed throughout the space around the receiver. An SNR of -70 dB translates to a requirement of a 12 bit A to D considering only the quantization noise. This figure indicates that our requirements of the A to D could be met by using off-the-shelf components. As an example, the AD9467 [26] has a specification of 75.5 dBFS when operating at 250 MSPS. However, for the case when tracking pulsars with an SNR poorer than -70dB, current ADCs would find it difficult to provide sufficient sampling speeds that are in turn required to process bandwidths for greater SNR improvement by the process of dedispersion, Section 4.5. This limitation could be circumvented by making use of a structure where we have a bank of parallel ADCs sustaining a larger effective sampling rate when the sampling time of each ADC is offset from the other.

3.1.5 Coherent or Incoherent dedispersion?

The dedispersion process stage is one of the fundamental blocks in the pulsar detection procedure. It is a major contributor both towards increasing the SNR of the pulsar signal as well as generating a stable pulse profile.

One of the key constraints towards devising the navigation system is the very poor input SNR of the pulsar signal. From Sections 4.4 and 4.5, we can note that the process of folding alone is not sufficient to detect the pulsar signal whenever a body is undergoing non linear motion. Dedispersion needs to at least provide an SNR improvement of the order of 50 to 75 dB, Section 4.5.

Coherent dedispersion allows one to obtain a better SNR improvement than incoherent dedispersion. This is because when one uses incoherent dedispersion, the comprising individual frequency channels themselves contain an inherent delay which inhibits precise dedispersion. Also, incoherent dedispersion is not suitable to process wide signal bandwidths at lower frequencies (suffering from large dispersion) which are in turn required to generate a large SNR improvement.

When it comes to generating a stable pulse shape, although the incoherent technique results in a stable pulse, however the probability of a pulse shape being unique to a particular pulsar diminishes. Incoherent dedispersion will not be able to produce the actual pulse shape accurately. Examples of pulse shapes generated using incoherent dedispersion can be found in [27] and [28].

Hence, considering the above factors, one should implement the technique of coherent dedispersion at the cost of relatively more computational resources than incoherent dedispersion.

3.1.6 Fast Folding Algorithm applicability

The Fast Folding Algorithm (FFA) was originally proposed to detect noisy periodic pulses present in a given set of data [29][30]. This algorithm is used in radio observatories to search for pulsars. In order to detect the presence of a pulsar, the dedispersed data is first stored, and is then processed offline using the algorithm. By following this method, one can reduce the number of computations that are required to fold at different periods.

The algorithm is best understood with the help of Figure 3.2, where a series of twelve elements are being folded at different periods. The FFA always produces multiple folds between a range of periods. The lower bound of the range can be a user specified period

P_0, but the upper bound is always P_0 added to the sampling time (T_s) of elements input to the algorithm. Hence, the upper bound would fold one additional sample than the lower bound. The intermittent periods between the two bounds depend upon the sampling time as well as the ratio between the number of samples that we store and process using the algorithm (N) and the number of samples comprising the lower bound of the range (P_{0N}).

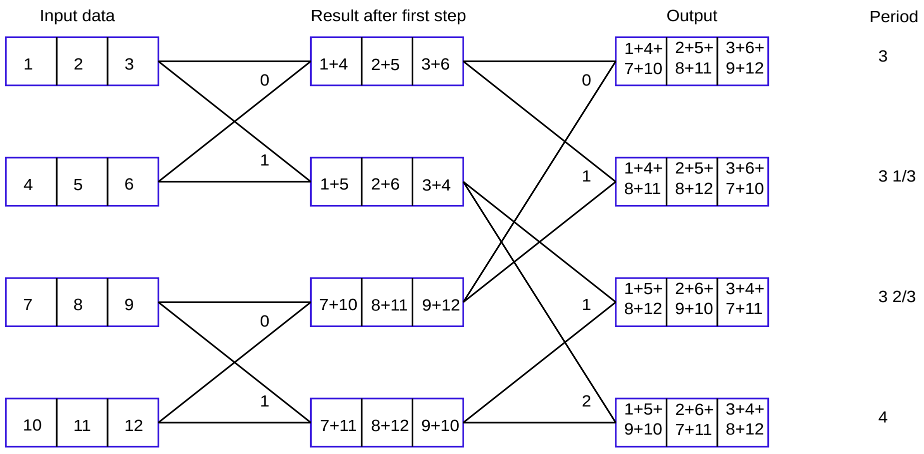


Figure 3.2: The FFA data flow

Numerically, the relation between the intermittent period resolution (R) at the output and the other parameters is given by [30]

$$R = T_s/((N/P_{0N}) - 1) \,. \tag{3.1}$$

The FFA however would be of little use in our navigation system. This is because of two reasons. The first is that the algorithm performs folding at periods that are uniformly spaced between two bounds, and is actually not tunable to perform folding at the real time requisite velocity dependent periods. The second reason is that the algorithm would produce approximate results as is explained below.

Since the upper bound is only greater by a single sample to the lower bound, one would need to club several samples together and treat them as a single sample in the FFA. Only this would allow us to produce periods folded at several different number of integer samples. However, this would cause a reduction in the accuracy of the output folded profiles, which would lower the correlation sensitivity of the subsequent block.

3.2 Proposed Architecture

3.2.1 Modified front-back end communication architecture

From our discussion in Section 2.6, we need to modify the architecture to a more efficient one. The architecture presented in Figure 3.3 reduces the number of computations that one needs to achieve the goal of pulsar detection. Here, the dedispersion is performed without Doppler correction. This modification is allowed since the receiver is now observing an input band of frequencies that are dispersed by the ISM at the same frequencies observed relative to the receiver. In this architecture, apart from the observed period, the front end transmits the current receiver velocity to the back end as well. This will enable a simple back end that makes use of velocity integration to estimate the coordinates of the body.

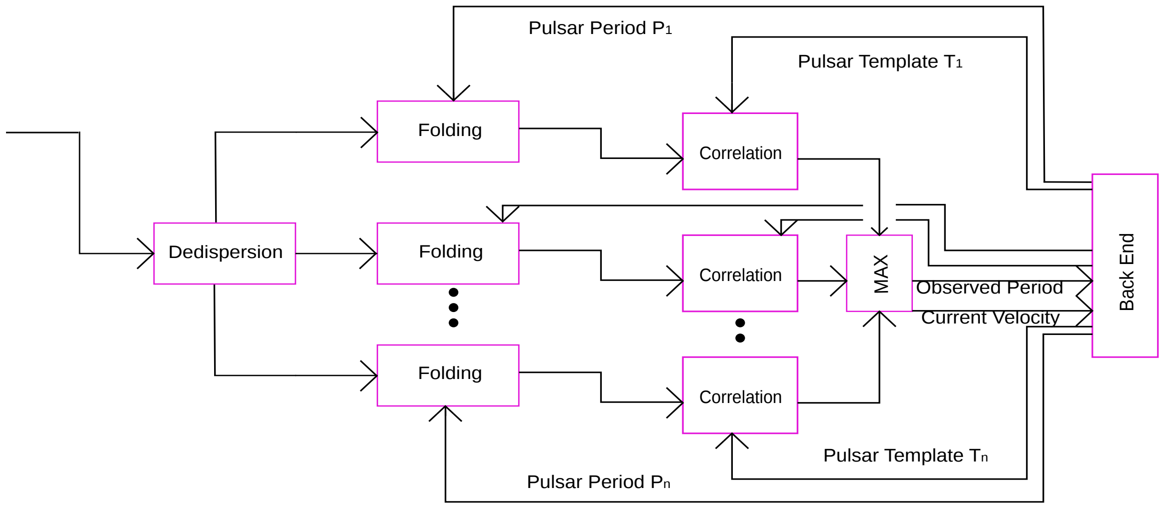


Figure 3.3: Modified communication architecture

3.2.2 The implementation possibilities

The choices of processing steps towards constructing the architecture of the signal processing front end have been considered, and are presented in the flow chart of Figure 3.4. In the flow chart, there are two different types of filters used namely the BPFB or Band Pass Filter Bank and the PFB or Polyphase Filter Bank. The analog domain filtering should take place using the BPFB. We should make use of FFT and IFFT banks when the filtering proceeds via the BPFB. This would allow a better dedispersion as each channel at the output of the BPFB would have a finite bandwidth which is then allowed to be processed by an individual FFT. Subsequently in such a case, the IFFT results of different channels can be added together to complete the dedispersion process. Also, one can in hardware use the method of resource sharing for the IFFT core amongst different channels at the cost of memory.

The Incoherent dedispersion in time domain proceeds via suitably delaying frequency sub-bands and then adding them together. Coherent dedispersion in time domain can be performed by convolving the input data with the inverse of the ISM transfer function. In this case, the BPFB should act as a BPF only. Correlation can be performed both in time, as well as the frequency domain. For correlation to be undertaken in the frequency domain, one needs to first take the FFT of the folded signal, and then perform a multiplication of the result with the complex conjugate FFT of the pulsar template. At the moment, additional work is required in determining the best correlation approach for our use.

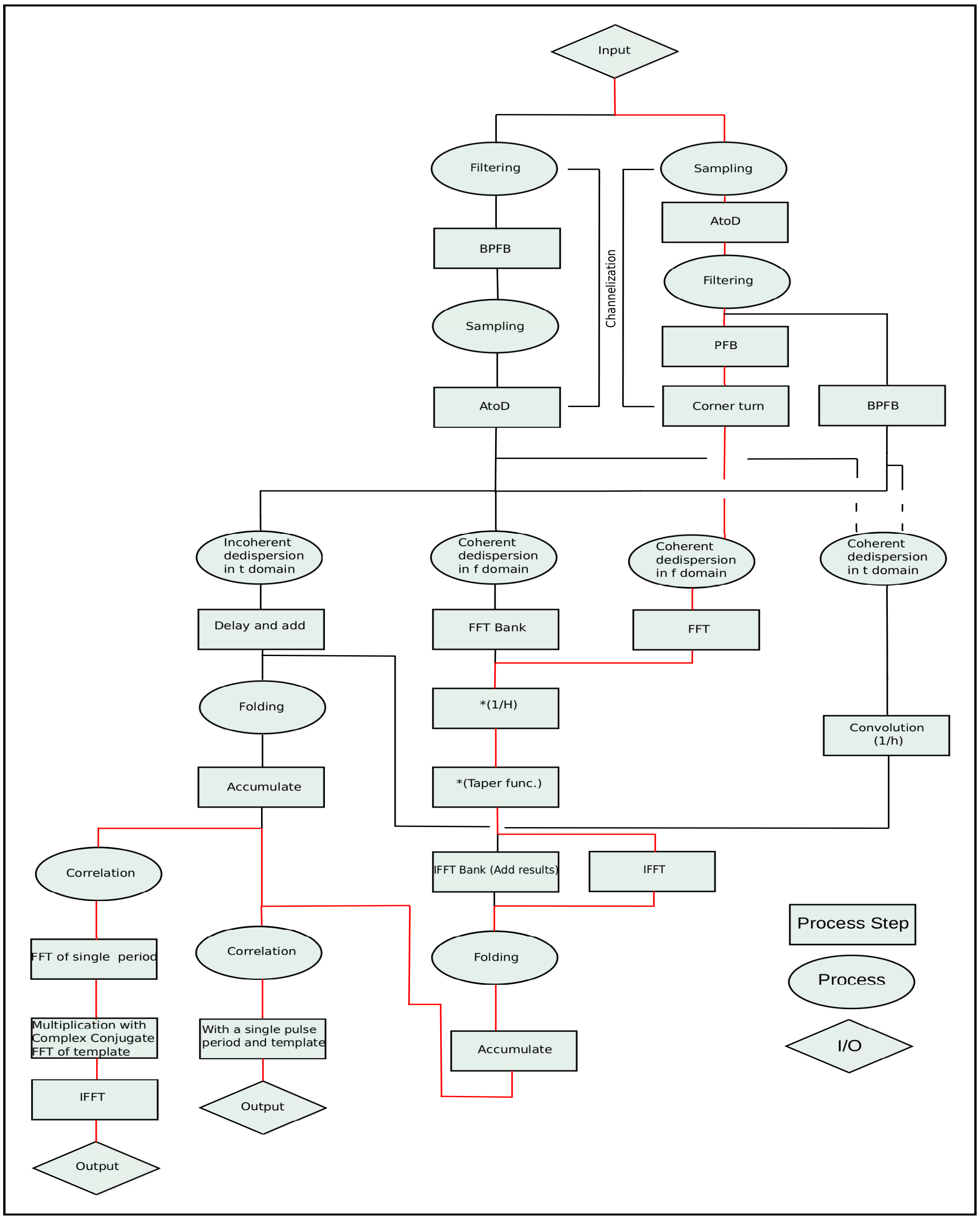


Figure 3.4: Implementation flow chart

The path in red denotes our intended line of design. In our approach of the hardware implementation, we use a PFB that allows us to produce channelized outputs. Following this is the 'Corner turn' as in [31]. Here, the PFB channel outputs are stored in a memory matrix. These are then read in a transposed manner and are input to the FFT core as shown in Figure 3.5.

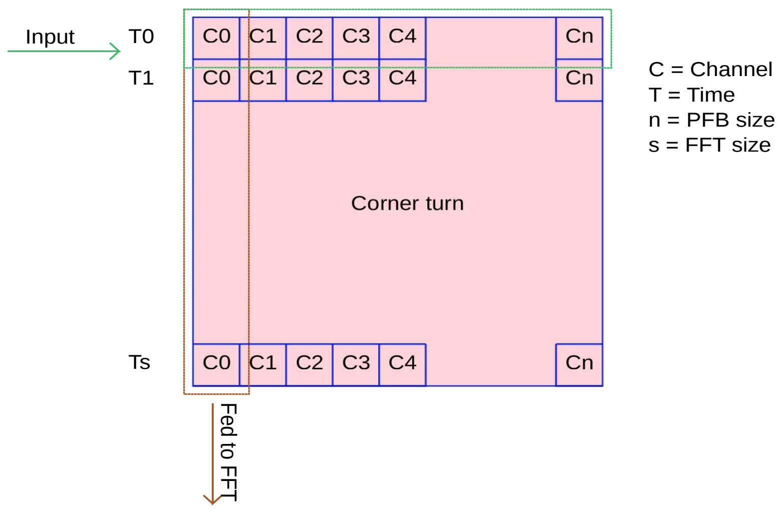


Figure 3.5: Corner turn

The inverse filtering procedure needs to be performed by the technique of overlap-save discrete convolution [5][17]. The input data length that needs to be convolved with the inverse ISM transfer function should be at least as large so as to cover the dispersion delay in the input band. One also needs to pad the start and end of the data with additional samples as required by the technique of the overlap-save convolution. Once the data is multiplied by the inverse transfer function, we process it with a taper function to emphasize anti aliasing [5].

After taking the IFFT, we need to discard the additional padded samples leading to the required dedispersed time series. Subsequently, the folding operation is performed in the time domain. Following correlation, we complete the front end procedure as described in Section 2.6.

3.3 Conclusion

We conclude that the signal frequency most suited for navigation depends on the location of the observed pulsar in the Galactic plane and our chosen dedispersion procedure. Next, the filtering process step of our system has been set to be implemented in the digital domain. Also, we find that current A to D converters provide satisfactory specifications to be useful for our application. Based on our requirement of obtaining a greater SNR improvement and distinct pulse profiles, we have concluded to make use of the coherent dedispersion technique. We analyze that the FFA would not be of use to fold pulsar signals for navigation. Furthermore, we can now make use of a modified architecture allowing a single dedispersion of the input frequency band for a particular pulsar. Finally, the signal input to the front end would pass through an A to D converter followed by a PFB, Corner turn, FFT and others.

4 SNR Enhancement Analysis

4.1 Approach

This chapter analyzes the SNR improvement offered by subsystems present in the signal processing front end. As had been mentioned in [4] that dedispersion should be implemented incoherently, we first find out how much SNR improvement can the folding subsystem provide. This would allow us to determine the SNR enhancement that dedispersion must provide and help us correctly choose between using coherent or incoherent dedispersion. The analysis of SNR enhancement by folding is done for different navigation scenarios presented in Section 1.2. We then discuss the SNR that is required for a successful correlation between pulsar signals and templates. After this, we estimate the required enhancement by dedispersion, and suitably present the bandwidth that is needed to be processed to achieve the necessary SNR levels. Finally, we propose the dedispersion technique that is necessary to be made use of.

Folding can be performed in both time as well as the frequency domain [5]. However, since for small antennas the pulsar signal will have a poor SNR, we can benefit from this process step only in time domain. This is because in the Fourier domain, the input power of a pulse with a small duty cycle is spread amongst various harmonics. Moreover, computing a finite size Fourier transform reduces the SNR of the signal from that which was input to the transform. Finally, the time domain approach maximizes the phase coverage in pulsar searches [5] as compared to the Fourier approach.

The following two sections address the question as to how much does the SNR increase for the case when folding is performed on a moving body. The analysis is carried out in line of our goal that the navigation system should be useful for deep space navigation as well as for use on-board ships and rovers on planets.

4.2 SNR Enhancement by Folding for Unknown Constant Receiver Velocities Relative to Pulsars

4.2.1 Priori

The analysis for a constant receiver velocity relative to pulsars should find practical use in deep space navigation wherein a spacecraft will undergo straight line motion. A simplified model of a received pulsar signal (sans noise) that we would be initially using for our analysis can be seen in Figure 4.1. We can note the width of the signal to be W, and the period to be T_0.

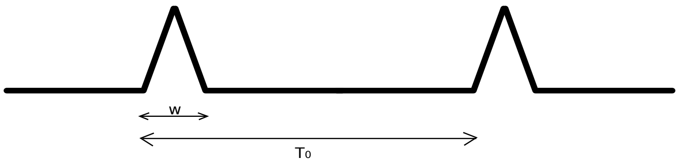


Figure 4.1: A simplified pulsar signal

Pulsars are moving sources themselves and there are several milli-second pulsars that are part of a binary star system where the two stars are orbiting around each other. Hence, by Doppler equation we have [3]

$$f = f_0(\frac{c \pm v_r}{c \pm v_p}) . \tag{4.1}$$

Here the symbols denote the following:

f = The observed frequency.
f_0 = Frequency of the emitted signal, or the inverse of the original period $1/T_0$.
c = The speed of light = $3 * 10^8 m/s$.
v_r = Speed of the receiver with respect to an inertial coordinate system.
v_p = Speed of the pulsar with respect to the same inertial coordinate system.

The modeling and correction of the pulsar velocity is in the scope of the signal processing back end system. Since at present no information is available on the pulsar velocity model, the velocity has been substituted as zero. This would indeed be equivalent to the case for pulsars that are moving in a straight line with a constant velocity. Accordingly, we derive the following formula

$$f = \frac{f_0(c \pm v_r)}{c} . \tag{4.2}$$

A receiver has the possibility to search for pulsar sources in its line of sight. Searching for a large number of pulsars at unknown velocities would need a great number of computational resources. Hence, it can be a good approach to limit the different possible receiver velocities used for searching pulsars as discussed below.

It can be a good approximation to consider only one of either positive or negative receiver velocities relative to the pulsars. In the case of a moving receiver, both the options have their own advantages in different scenarios. For the case when the receiver maintains its straight line motion for a prolonged duration, we should consider negative receiver velocities relative to pulsars. In other words, we should search for pulsars that are in the opposite direction of motion. This is because, such velocities would result in a longer observed Doppler period allowing a greater number of folds as would be clear in the next section. If however the body would change its state of motion after a certain interval of time, it would be better to search for pulsars in front of the body. This would result in a smaller Doppler period or a larger Doppler frequency and would contribute to more number of folds in a given limited time.

Following such an approach will allow us to reduce the velocity search space and the computational resources by a factor of two. Also, it will allow us to use an antenna that would only need to search for pulsars in half of the visible sky in the case of a ship or rover navigation, or 25% of the total sky surrounding the spacecraft.

Based on the above discussion, let us assume that the receiver is moving away from the source and maintains its line of motion for a prolonged duration such as in spacecraft navigation. Hence v_r would be subtracted in the above formula in accordance to the Doppler effect.

4.2.2 Derivation

Let the period at which we are folding have a difference of Δ from the observed period. Hence Δ is found by the following equation

$$\Delta = T_{observed} - T_{folding} = T_0(1 - \frac{v_r}{c})^{-1} - T_0 \approx \frac{T_0 v_r}{c} \; . \tag{4.3}$$

Due to there being a difference in the received and folded period, we will attain an SNR improvement that is less than for the case when folding is performed at exactly the same period. This deviation will be quantitatively analyzed for later in this section. Also, when folding is not undertaken at the actual period, we need to take care that we do not miss the continuity of the pulse profile while folding. This will be discussed in Section 4.3.1.

Let the number of samples in a period of the emitted pulsar signal be x. As is mentioned before in Section 2.6, to determine a particular period we would be folding at other periods (in order to detect the apparent or received period due to the change by Doppler effect). In the hardware realization this would translate to performing the folding operation for periods having a different number of samples than x. The minimum Δ depends on the resolution, in other words the closest period at which folding is performed. To minimize the error between the actual input period and that which is being folded for, we must have a very fine folding resolution or a very small difference between the periods for which we are performing folding. Let folding be performed for a possible received period that contains a total of $(i + x)$ number of samples per period. Here i denotes the additional number of samples in the period for which we are performing folding as compared to the sans Doppler effect period emitted by the pulsar. The domain for i depends on the resolution of our folding algorithm. Let this folded period be the closest to T_0.

Hence, now the minimum Δ will have the value,

$$\Delta_{min} = T_0(1 - \frac{v_r}{c})^{-1} - (T_0 + \frac{iT_0}{x}) \approx T_0(\frac{v_r}{c} - \frac{i}{x}) \; . \tag{4.4}$$

When $\Delta_{min} > 0$, the peak shifts with respect to the intended sample of folding. After N folds (N_{folds}), the accumulated 'residue' (R_{accum}) or the net shift will be,

$$R_{accum} = \sum_{i=1}^{N_{folds}} \Delta_{min} = N_{folds} * \Delta_{min} \; . \tag{4.5}$$

The number of folds contributing to an increase in SNR (by amplitude) are found as presented next.

Let the average dimensions of a received pulsar signal be as shown in Figure 4.2.

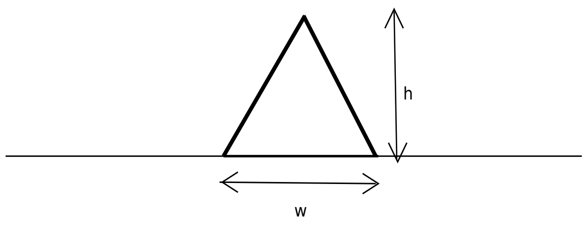


Figure 4.2: Signal dimensions

In reality there are deviations to the above assumed signal dimensions. The received signal from a pulsar varies amongst individual pulses. However, the integrated pulse shape remains the same between different observations for a pulsar. The observed integrated pulse shape depends on the emitted beam from the pulsar as well as the angle between the receiver and the beam center [5]. There are pulsars that emit a double peak structure and more complex shapes. A sketch of two possibilities is presented in Figure 4.3.

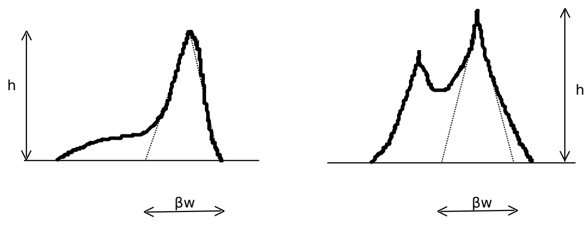


Figure 4.3: Complex pulsar shapes

These variations can be coarsely approximated by a unique (pulsar specific) scaling factor β of the width of the pulsar signal. Such an approximation would result in an SNR improvement that would be less than that which would be actually obtained. Once the pulsar candidates for navigation are determined, β can be given a common minimum value amongst all the known profiles for analysis. Also due to there being a Doppler shift, the pulse width would either increase or decrease, i.e the wave would be compressed or would undergo rarefaction. However, for the current analysis β as well as the change in the width received due to Doppler would be assumed as unity and these changes have been corrected for in the final expression for the SNR improvement presented towards the end of this section.

Now, the maximum number of folds that would contribute to the improvement in SNR would be till the time the received pulse overlaps the samples of the first pulse present in the accumulator. This is in line of the afore mentioned fact that the period would be in time shifting from its intended sample position because of a difference in our folding period. The number of folds for the case when the period of folding is smaller than the received period is given by

$$\gamma_1 = \frac{w}{\Delta_{min}} = \frac{w}{T_0(v_r/c - i/x)} \ . \tag{4.6}$$

As in the above formula the received period is larger than the period with i additional samples, we must also check if the next folding period which is larger than the received period has a smaller Δ_{min}. We must calculate

$$\gamma_2 = \frac{w}{T_0((i + Res)/x - v_r/c)} \ . \tag{4.7}$$

Here, Res denotes the resolution, and is the difference in the periods at which we are folding. Now we select the maximum out of γ_1 and γ_2 and let it be denoted by γ. Referring to the previous model, a shift in Δ_{min} will appear as seen in Figure 4.4.

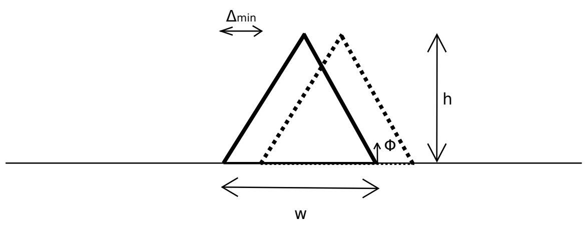


Figure 4.4: A shift in the signal position during folding

Hence, while the number of folds are less than or equal to γ, we would in time attain an increasing signal amplitude. The new peak would be at a different location than $\frac{w}{2}$. The maximum sum in the situation of a shifting signal would be attained at the marked point ϕ, or the edge of the ideal pulsar signal.

The resulting pulse shape would be different from the one which is input. Also, if folding is performed for the duration $> \gamma$, we would observe a peak formation in adjacent samples and a flat peak top would start developing. This has been verified using simulations in Matlab.

The obtained signal amplitude (A_{signal}) can be found by the following formula

$$A_{signal} = 2 * \frac{2h}{w} * \sum_{u=\Delta_{min}}^{\frac{\gamma\Delta_{min}}{2}} (u) \ . \tag{4.8}$$

Where u takes the values Δ_{min}, $2 * \Delta_{min}$ and so on. The above expression on simplification results in

$$A_{signal} = h * (\frac{\gamma}{2} + 1) \ . \tag{4.9}$$

Thereby, the effective number of folds $N_{effective}$ is given by

$$N_{effective} = (\frac{\gamma}{2} + 1) \ . \tag{4.10}$$

On correcting for a change in width due to β and Doppler, one acquires an expression of the form

$$N_{effective} = \frac{\beta\gamma}{2(1 - v_r/c)} + 1 \approx \frac{\beta\gamma}{2} * (1 + \frac{v_r}{c}) \,. \tag{4.11}$$

The improvement in SNR (by amplitude) due to folding is given by the relation $\sqrt{N_{effective}}$ as is presented in [18] and is said to follow boxcar averaging. Hence, the SNR (by amplitude) is improved by a factor of

$$\sqrt{\frac{\beta\gamma}{2} * (1 + \frac{v_r}{c})} \approx \sqrt{\frac{\beta\gamma}{2}} \,. \tag{4.12}$$

4.2.3 Numeric examples

To get an idea of the order of improvement in SNR (by amplitude) for a spacecraft moving at a constant velocity, a simulation has been performed including the Doppler effect. For a pulsar period of 1 second, sampling rate of 1 MHz, spacecraft velocity of 17,000 m/s, a folding resolution of 10,000 samples, and an average β value of .5; we obtain an SNR improvement of approximately 70 dB. The time it takes to obtain such an improvement is nearly 475 days. As another example; when folding at exactly the same period as that input to the system, the SNR is enhanced by 70 dB in 116 days. Even though the SNR improvement that is obtained is large enough for us to detect pulsars, the time it takes to improve to such levels is considerably high. This indicates that we must also adopt dedispersion to enhance SNR for such scenarios.

The following section elucidates the SNR improvement (by amplitude) for the case when the velocity of the body as seen by the pulsar is not a constant.

4.3 SNR Enhancement by Folding for Unknown Variable Receiver Velocities Relative to Pulsars

4.3.1 Priori

The case of an unknown variable receiver velocity would be encountered in scenarios such as ship navigation on Earth and rover navigation on other planets. Hereafter in this section, the term Earth represents a generic planetary body.

Such a scenario needs to be considered as the Earth would appear to be accelerating to stars because of rotation about its axis as well as its orbital motion around the Sun. Although we know the velocity of Earth as a function of time in its orbit, the linear velocity of a body with respect to the pulsar would depend (at an instant) on its position (latitude and longitude) on Earth (because of rotation). This can be understood in Figure 4.5 presented next.

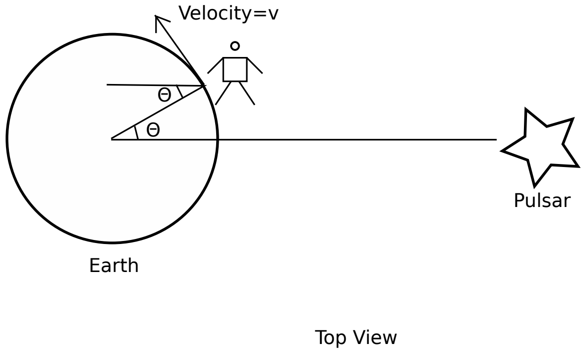


Figure 4.5: The relative velocity depends on the receiver location

Here θ represents the unknown longitude with respect to the direction of the pulsar from the current Earth location in space. The magnitude of velocity in this direction can be found by multiplying the velocity with $\sin\theta$.

The linear velocity of a body is highest at the equator and reduces as a function of its latitude (assuming Earth to be a uniform sphere). It can be understood by the help of Figure 4.6.

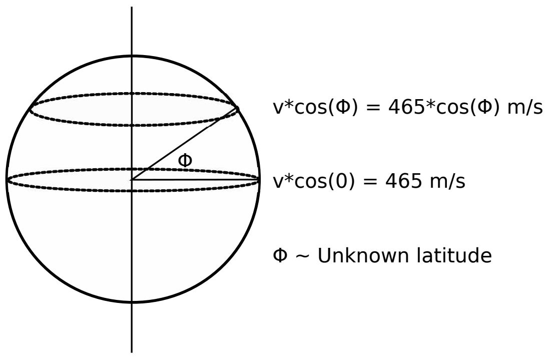


Figure 4.6: The linear velocity at a point on Earth as a function of its latitude

Thus far the velocity considered is only due to the rotation of Earth. Extending to accommodate the velocity of the body will simply result in

$$v = \pm v \sin\theta \cos\phi \pm \overline{v_b}\,. \tag{4.13}$$

where $\overline{v_b}$ is the component of velocity relative to the pulsar.

Here again $\overline{v_b}$ would affect θ and ϕ but this would be analyzed for later. At this point it would be good to review that we would be folding at different periods to accommodate variations because of the Doppler phenomenon. The Doppler effect would result in compression/rarefaction of the pulsar signal. When folding is not performed at the exact period, two constraints need to be taken care of

1) The pulsar signal should not appear at the edge of the frame in the accumulator as this would result in a loss of the true pulse shape in subsequent foldings.

2) The effective velocity of the body should increase or decrease only in one direction and that too with constant acceleration. When a body undergoes variable acceleration; the pulsar signal's SNR would increase in the process of folding which could be performed for a longer duration irrespective of the velocity at which we are folding. However, we will attain in such a situation an uncorrelatable resulting pulse profile.

The first constraint maybe corrected by conducting another folding in parallel (henceforth the two parallel folds implemented would be denoted by P_{folds}) which skips a certain number of samples as compared to the first one. The number of samples skipped should be greater than the number of samples contained in the pulse portion of the signal and less than the total number of samples in a period of the signal by the same number. This would ensure that there is no loss of the pulse while folding. The second constraint maybe corrected by making use of an augmented accelerometer system. Moreover, the second constraint might also be alleviated if the system before folding provides a high SNR which would allow us to speed up the velocity estimation capability of the system.

4.3.2 Derivation

Now, v_b can be written in terms of its latitude and longitudinal components. Clearly, $\overline{v_b}$ = $v_b \sin(\theta_2)\cos(\phi_2)$. Here θ_2 is the longitudinal component of the velocity of the body. Likewise, ϕ_2 is the latitudinal component of the velocity of the body.

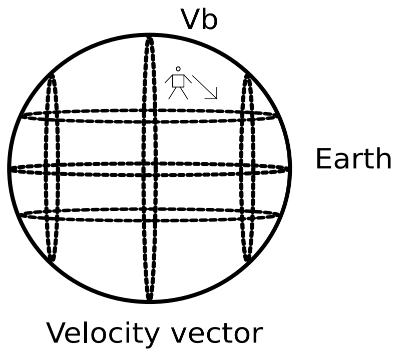


Figure 4.7: The velocity vector of a body on Earth

The velocity derived thus far is not the actual velocity relative to the pulsar, since it itself is at an elevation relative to the Earth (or the receiver antenna). The elevation with respect to the Earth is known and has been recorded relative to the Earth's tilt. It is commonly called 'declination'. Hence on taking this into account we would also be compensating for the change in velocity due to the angle subtended to the pulsar by the Earth's tilt.

Since V is the net velocity in the line of sight of the pulsar, we can find the actual observed velocity by the relation

$$V = (v \sin\theta \cos\phi \pm v_b \sin\theta_2 \cos\phi_2)\cos(\theta_{declination})\,. \tag{4.14}$$

Based on our understanding from Section 4.2.1, for the scenario when a body moves with a velocity that is changing direction in time, we should search for those receiver velocities that appear to be positive relative to the pulsars. In order to simplify the appearance of expressions, the remaining analysis would be undertaken for a body with maximum speed values equivalent to the highest speeded sea ships in use today. For any other analysis, v_b should be appropriately given a maximum value.

The fastest naval ship has a speed of around 18 m/s. Let the maximum speed of water in the ocean be equal to 5 m/s. Thereby the maximum speed of a ship will be around 23 m/s.

A pulsar would observe the maximum velocity of a body in the case when: it is at 0 declination, the latitude of the body is 0 and, its longitude on Earth relative to the direction of the pulsar is 90. Also, the velocity of the body should have the same direction as that of Earth at that position. In other words, the velocity of the body makes a zero degree angle along the line of sight to the pulsar as is shown in Figure 4.8.

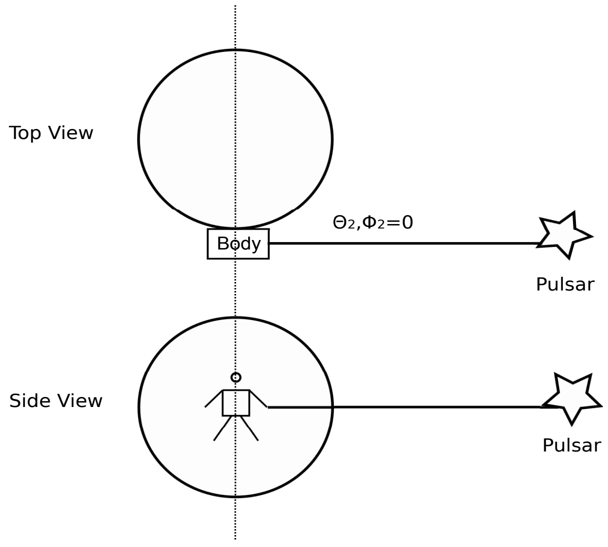


Figure 4.8: Line of sight at maximum relative velocity

We have, 465 m/s as the rotational velocity of the Earth about its axis at the equator. Therefore, the range of velocities observed by the pulsar equal -488 to 488 m/s. One must note that the worst case declination of 0 has been taken to be true for now and can be suitably changed once more information is available on the pulsar database.

The back end system which would be using Kalman filter based signal processing algorithms to compute the coordinates of the body would require a specific velocity resolution from the hardware preceding it for precise coordinate estimation. Let this required resolution in determining the velocity = ‘R’ m/s. Hence, the angle elapsed while changing from one velocity to the other velocity that is measured (assuming an acceleration due to Earth) is equal to

$$\alpha = \cos^{-1}(1 - \frac{R}{488}) . \tag{4.15}$$

It is to be noted that in order to prevent a change in sign of the velocity of the body in the Doppler equation and hence prevent the case of variable acceleration observed by pulsars, there should be at-least more than one pulsar observing for each dimension of velocity. Moreover, the two pulsars should be separated by at-least the angle derived above. Hence we will now need a minimum of six signal sources (pulsars) for navigation.

A body resting at the equator takes 24 hours to rotate about the Earth's axis by 360°. Using ratio and proportion, the folding operation can be performed for

$$T_{fold} = 24 * (\frac{465}{488}) * (\frac{\alpha}{360})\ hrs . \tag{4.16}$$

This time is not limited by the situation wherein the input pulses to the folding subsystem do not overlap with the first pulse present in the accumulator. This is because, for such velocity ranges T_{fold} is much less than the time for which we can perform folding at constant velocity even without any parallel folding to compensate for the Doppler effect.

Let the body have a net acceleration of a. Since the acceleration by a body can change its velocity dramatically within a short period of time, we can initially target for the case of a body continuing in a state of uniform motion with respect to Earth. Thereby, the effective acceleration would only be because of the Earth's inherent acceleration with respect to the pulsar. Hence the period observed at time t (time step = T_0) with the k^{th} fold in progress is given as

$$\frac{T_0}{1 + V_k/c} \approx T_0 * (1 - \frac{V_k}{c}) . \tag{4.17}$$

Here, V_k is the new velocity seen by the pulsar as the body on Earth is now moving at an angle to it. Its value is found from vector geometrics.

Following an earlier discussion, the observed width at the k^{th} (arbitrary) fold would have a value of

$$\frac{w}{(1 + V_k/c)} \approx w * (1 - \frac{V_k}{c}) . \tag{4.18}$$

The maximum value of k, limited by the velocity resolution, is found by using the relation

$$k_{max} = (\frac{T_{fold}}{T_0})\ hrs . \tag{4.19}$$

One must note that in such cases the residue is generated because of a difference in the observed periods due to changing velocities as well as a difference in the pulse widths at subsequent periods. Thereby, at each subsequent fold, the shape of the pulsar would appear different. The accurate residue generated at the k^{th} iteration would be given as

$$\Delta_{k^{th}} = abs\left(\frac{T_0 * c * (V_k - 488)}{(c + 488) * (c + V_k)}\right) . \tag{4.20}$$

Since the maximum value of k as presented above is a variable unless calculated for a particular case, one cannot determine the resulting SNR without undertaking simulations that would take into consideration the maximum shift from the original intended position.

4.3.3 Simulations

For milli-second pulsars the pulse period which would lead to the lowest SNR on folding is 1 second. This is indeed the case as verified by simulations. The graph of Figure 4.9 is a plot of increase in SNR (by amplitude) on folding against the period. The sampling rate has been chosen as 200 KHz. For all the subsequent simulations, the width of the pulsar has been chosen to be 1% of the total period. Also, the velocity of the body was chosen to be 51 m/s which is possibly a worst case scenario for use in commercial trucks and rovers.

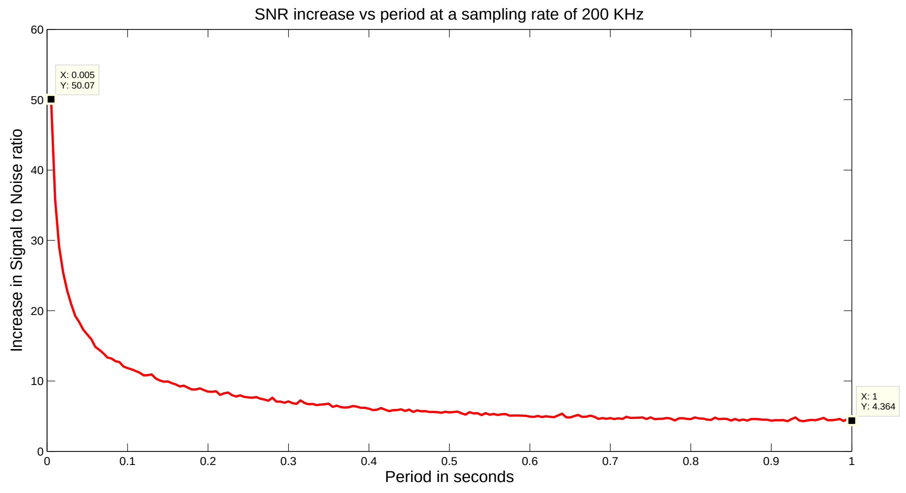


Figure 4.9: SNR improvement as a function of pulsar period

Since during the time of this thesis no information is available on the requisite velocity resolution by the back end, the resolution of .1m/s has been chosen as a best case scenario for improving the SNR. This because, a resolution of this magnitude would already translate to a huge error of 360 meters of distance in 1 hour if one were to follow simple integration based methods to calculate the position based on velocity.

Simulations have been performed to measure the SNR improvement (by amplitude) against the velocity of the body for different sampling rates. A suitable choice of the sampling rate can be made from the simulations. The choice would depend on the trade-off between the SNR improved and the computation power required in hardware. The effect of the velocity of the body on the SNR can be understood by Figures 4.10 to 4.12.

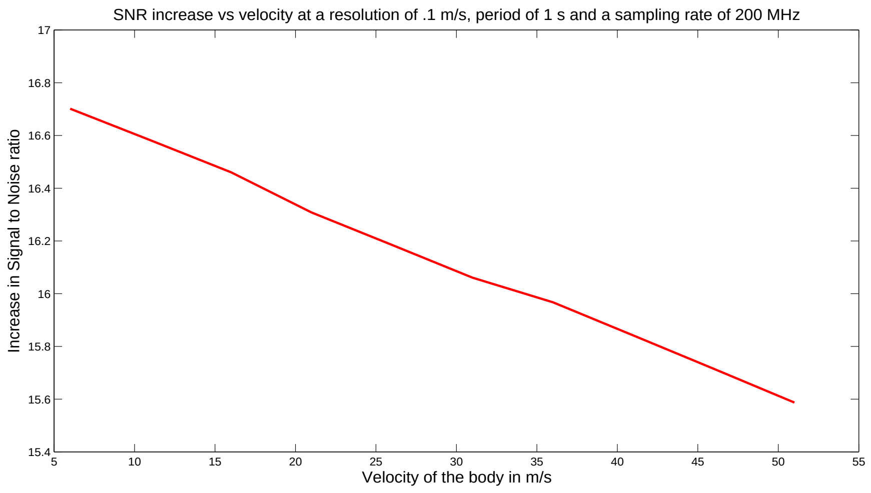


Figure 4.10: SNR improvement versus velocity at a sampling rate of 200 MHz

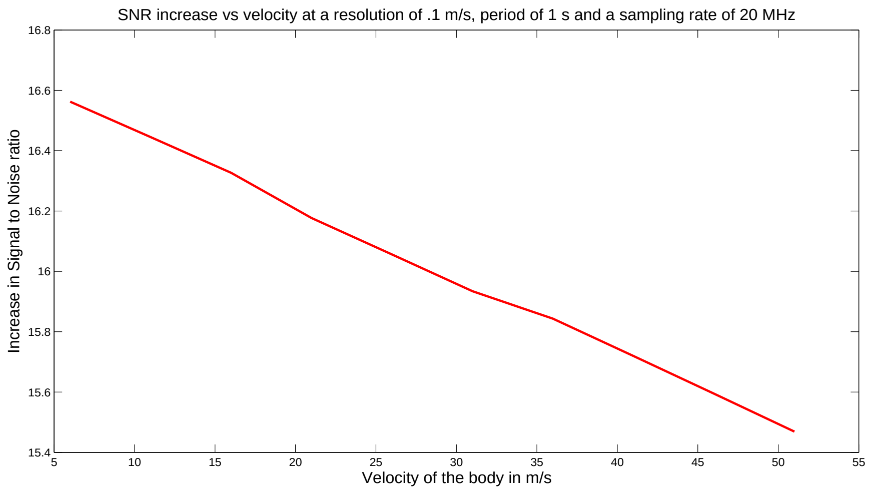


Figure 4.11: SNR improvement versus velocity at a sampling rate of 20 MHz

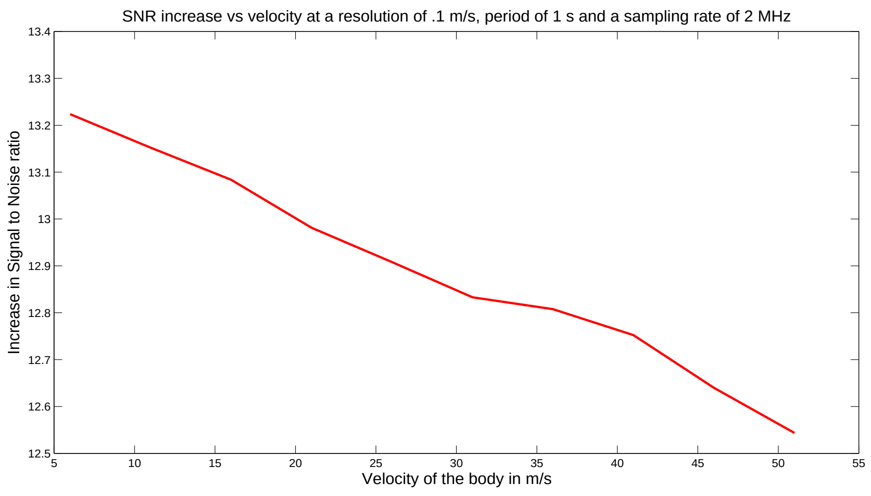


Figure 4.12: SNR improvement versus velocity at a sampling rate of 2 MHz

The above graphs indicate that the sampling rate can be chosen to be roughly around the 2 MHz mark. For the maximum assumed velocity of the body, a simulation is performed indicating the increase in SNR versus the sampling rate. We approximately obtain an SNR improvement by a factor of 10 (20 dB) on folding for a sampling rate of 1 MHz. We select this to be our folding sample rate. In the next two sections, we first discuss the SNR requirement of the correlation subsystem and then estimate the needed input SNR at the folding subsystem. Following this, we derive the SNR enhancement needed to be provided by dedispersion and then propose the necessary bandwidth and dedispersion technique to be implemented.

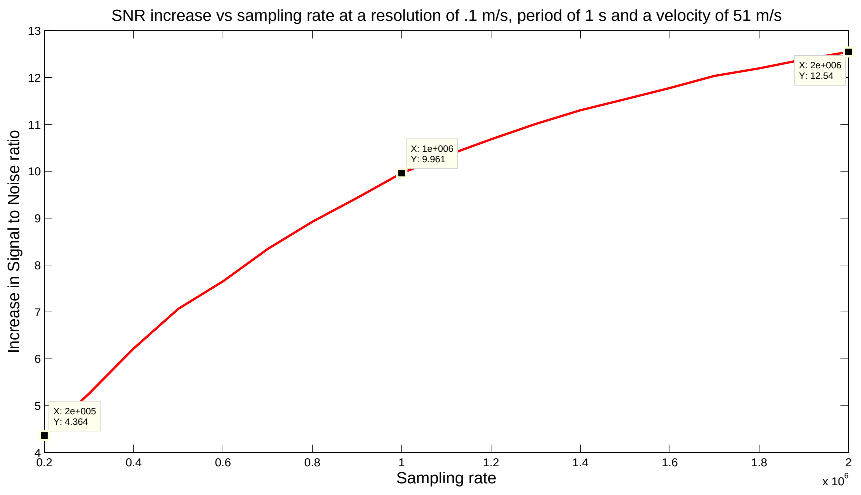


Figure 4.13: SNR improvement versus sampling rate

4.4 SNR Requirement for Correlation

This section helps us in our task of determining the SNR enhancement that is required to be achieved by the process of dedispersion. As was mentioned before, folding is a process step that helps to generate the stable signal profile which would subsequently be correlated with known templates. In order to determine the requisite SNR at the input of the folding block for generating a profile that can be correlated; we need to take into consideration the SNR value needed to perform a successful correlation as well as the improvement in SNR provided by folding. Once we know the necessary SNR after dedispersion, we propose the bandwidth that should be dedispersed in the next section.

As presented in the tutorial on 'Canonical Correlation Analysis' [32], the relation between SNR (S/N) and the correlation coefficient (ρ) of a signal and its noise corrupted version is

$$S/N = \frac{\rho^2}{(1-\rho^2)} \; . \tag{4.21}$$

By the above formula, the value for the correlation coefficient approaches 1 for an SNR of around 25 dB. A plot of the correlation coefficient versus SNR is presented in appendix B. As was discussed before in Section 4.2.2, the improvement in SNR because of folding is given by the square root of the effective number of folds. Hence, in order to perform correlation perfectly; we must have

$$20\log(InputSNR * \sqrt{N_{effective}}) = 25 \; dB. \tag{4.22}$$

Future work is actually required to pinpoint the lowest SNR that would allow a successful correlation for navigation. Intuitively, to be able to navigate with a high accuracy, we would need perfect correlations since the change in signals due to the Doppler effect would be very small. However, we must have an idea of what would be the navigation accuracy's dependence on the correlation coefficient and then chose a minimum suitable value. For now, we can assume a possible range of suitable SNRs. We can see in appendix B that an SNR of 0 dB corresponds to a correlation value of approximately .7. Thereby on setting the SNR range to be from 0 to 25 dB, we narrow ourselves to expected correlation values of .7 to 1.

We can note from Sections 4.2.3 and 4.3.3 that folding provides lesser improvement in SNR for the case when the receiver undergoes nonlinear motion with respect to the pulsar. It would be better if we design our system considering the latter scenario where we obtain an enhancement of 20 dB. What it means is we would be using the dedispersion process considerably to help raise the SNR to the required ranges. This is good because shifting the SNR enhancement load to the dedispersion block would speed up the processing time of the front end and in turn allow more accurate navigation. Moreover, our system would then find use in all the scenarios that we desire to navigate in.

Now, the input signal to noise ratio to the folding block (F_{SNR}) must equal

$$-20 \; dB \leq F_{SNR} \leq 5 \; dB \; . \tag{4.23}$$

4.5 Dedispersion Bandwidth

Just as in the case of folding where the SNR increases due to adding consecutive pulses to each other, in dedispersion the different frequency components are aligned and added together. After dedispersion, the SNR should increase by the square root of the amount of frequencies processed. Indeed, it is commonly found in literature that the SNR enhancement by pulsar receivers is directly proportional to the square root of the observed frequency band. We can infer that when dedispersion is applied, the theoretical SNR enhancement (D_{SNR}) by amplitude equals [5][18]

$$D_{SNR} = \sqrt{BW}\ . \tag{4.24}$$

Here BW denotes the bandwidth of the pulsar signal that we are processing. From the previous section, the output of dedispersion should have an SNR between -20 dB to 5 dB. As stated in Section 3.1.4, the observed SNR after the A to D would be around -70 dB. Since we now know the SNR value at the input of the dedispersion block and that after it, we determine the enhancement that dedispersion must provide. The enhancement lies between 50 dB to 75 dB. One can find the corresponding required bandwidths from Figure 4.14. Since we need to obtain a very high SNR enhancement by dedispersion, process large bandwidths and generate distinct pulse profiles; we must make use of the coherent dedispersion technique.

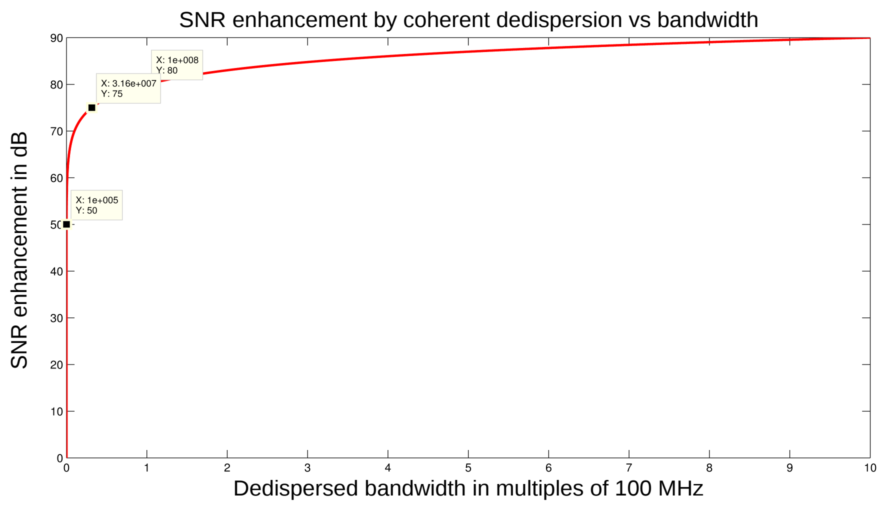


Figure 4.14: SNR enhancement by coherent dedispersion vs bandwidth

The above required bandwidth can be reduced by half if the system antenna allows for detection of two polarizations which can be averaged after the IFFT. Adding signals from two different polarizations will lead to an increase in the signal to noise ratio [18].

4.6 Conclusion

From this chapter we can conclude that SNR enhancement by folding alone is not sufficient to allow radio pulsar based navigation. Even though folding by itself allows pulsar detection when a receiver maintains a uniform velocity with respect to the pulsar, the time it takes to do so is too large for real world applications. On analyzing the SNR enhancement by folding for receivers on ships or rovers, we estimate that the improvement will be around 20 dB in magnitude. Further analysis revealed that dedispersion must provide enhancements around 50 to 75 dB and we are required to adopt the technique of coherent dedispersion to process pulsar signal bandwidths nearing a value of 30 MHz.

Hardware for Real Time Implementation 5

5.1 Approach

In this part of the thesis, we present our work of determining a suitable hardware platform for implementing the folding subsystem in real time. We begin with a calculation of the computational requirements. This is followed by a presentation of the architecture as should be implemented on an FPGA board. We then estimate the feasibility of an FPGA based implementation, and subsequently propose the needed hardware.

5.2 Computational Requirements

5.2.1 Unknown initial receiver position and velocity direction

This section addresses the required capability of hardware to perform the folding operation for the case when we do not know the initial coordinates and velocity direction of the receiver with respect to the celestial body on which it is traveling. The coordinates of the celestial body however must be known to allow navigation.

In order to calculate the required hardware we must find out the number of pulsar sources which we will be using to navigate. A larger number of pulsar sources will translate to a greater requirement of dedicated hardware. Since the number of pulsar sources needed for navigation at a particular instant of time is six (Section 4.3.2), nearly 50 pulsars should be searched for as explained next.

This number is an outcome of taking certain factors into account. The first being an assumption that pulsars are uniformly distributed throughout space. The second being the receiver will be navigating using 25% of the sky at any given instant of time (Section 4.2.1). Considering these two factors results in the requirement of 24 pulsar sources to be searched for. Now, the pulsar signal source reliability is an unknown at present. This reliability can be increased by using say, twice as many pulsars for navigation; resulting in a total of 48 pulsars. In the final implementation however, once more information is available about the source reliability; we can can appropriately scale the number of pulsar sources required for navigation.

An essential parameter that is needed to construct the folding subsystem is the number of bits (B) that are used to represent a sample during the folding operation. The number of bits will determine the range of signal values that can be processed by the folding subsystem using fixed point arithmetic. In the process of folding, a real time input sample is added to a sample already stored in an accumulator. The word length required to represent the sample value present in the accumulator depends on two factors. The first is the maximum value that the process of folding can result in, which requires a specific number of bits to the left of the decimal point in fixed point notation. The second factor being the minimum signal value that needs to be represented in order to successfully perform folding, and determines the number of bits to the right of the decimal point.

The maximum value that needs to be represented depends on the input dynamic range to the folding subsystem as well as the duration of folding. For a folding duration of at most ten minutes (limited by the back end velocity resolution for the scenario of Section 4.3) and a dynamic range of 0 to 1 at the input of the folding subsystem, one requires a total of 22 bits to represent the maximum value. Also, to accommodate the bits that would be used to represent values smaller than unity; we must take care of the signal to noise ratio in a sample. The input SNR to the folding subsystem should at least be at a level that will allow successful correlation to detect the pulsar source and the velocity with which the receiver is moving with respect to it. From Section 4.4, the input SNR to the folding block needs to be approximately -20 dB to produce a correlation of nearly 70%. We need 4 bits to represent such a signal value. Thereby, the total number of bits required B equal 26.

The period of any milli-second pulsar which we may use should range between 1 msec to 1 sec. Hence the average probable period ($Period_{average}$) for all sources will be 500 ms. Now, the hardware resources needed to attain a particular SNR depend on a few parameters. The parameters include: the number of samples per second that are being folded (S), the number of bits that are used to denote each sample (B) in hardware, the maximum velocity with which the body maybe moving (V m/s), the requisite velocity resolution by the signal processing back end (R m/s), the number of parallel folds (P_{folds}), and the number of pulsars which we will be searching for (P).

The amount of hardware required is in direct proportion to every parameter except the velocity resolution required by the back end to which it is inversely proportional. This is primarily the case because we require dedicated hardware to implement folding for different pulsars at various possible velocity dependent observed periods. Now, the theoretical number of fixed point operations required per second to perform folding (Fi_{ops}) can be calculated from the following expression:

$$Fi_{ops} = \frac{S * V * P * P_{folds}}{R} \ . \tag{5.1}$$

In Formula 6.1, a unit operation that is being performed is the fixed point addition of two samples in the process of folding. From our analysis in the previous sections of this thesis, we can have numeric substitutions for the afore mentioned parameters. From Section 4.3.3, the parameter R is assigned a value of .1 m/s. As seen in the simulation graphs from Figures 4.10 to 4.12, the velocity of the body is set to have a maximum value of 51 m/s. Now, the total velocity of the body is found after adding the velocity of Earth (or that of any other similar celestial body) to this value, and hence the maximum velocity equals a value of 520 m/s approximately. P_{folds} has been discussed before in Section 4.3.1 and denotes a value of 2. Finally, the required number of pulsar sources has been put forth as 50 as discussed before in this section. Hence, we can now denote Fi_{ops} by the relation

$$Fi_{ops} = 5.25 * 10^5 * S \ . \tag{5.2}$$

The relation between increase in SNR and S observed in Figures 4.10 to 4.12 as well as Fi_{ops} by using Equation 5.2, is presented with the help of the following table:

S	Fi_{ops}	SNR enhancement (dB)
1 MHz	$5.25*10^{11}$	19.96
2 MHz	$1.05*10^{12}$	21.96
20 MHz	$1.05*10^{13}$	23.78
200 MHz	$1.05*10^{14}$	23.85

Table 5.1: Sampling rate vs Fixed point operations vs SNR enhancement

The total memory bandwidth ($MEM_{bandwidth}$) required to support the computations can be found by multiplying the Fi_{ops} with B as shown in Equation 5.3.

$$MEM_{bandwidth} = B * Fi_{ops} \; . \tag{5.3}$$

Plugging in the value of B, for a sampling rate of 1 MHz the required memory bandwidth has a value 1.70625 TB/s. This bandwidth is separately needed both for 'read' as well as 'write' operations. The size of the memory would be determined by the periods at which folding would be performed. For an average period value ($Period_{average}$) of 500 ms amongst the various periods that are being searched for; the total memory required is theoretically found by the relation

$$MEM_{total} = B * Fi_{ops} * Period_{average} \; . \tag{5.4}$$

At a sampling rate of 1 MHz, Fi_{ops} of $5.25*10^{11}$ and a $Period_{average}$ of 500 ms, MEM_{total} has a value of 853.125 GB. In the practical implementation of the folding algorithm, there maybe some difference in the memory specifications that are actually required as compared to those derived above. However, the current calculations help in estimating the order of resources that will be needed for a successful implementation.

The memory bandwidth that is required for this process to be implemented in real time is enormous by any standards. Hence, we need to limit our design to a scenario wherein the receiver's initial position and velocity direction relative to its host celestial body are known approximately at the start of navigation. This should allow us to reduce our computational requirements. The requirements are derived in the next section.

5.2.2 Known initial receiver position and velocity direction

In such a scenario, a receiver's position as well as its velocity direction are known approximately with respect to a reference system right at the start of navigation using radio pulsars. An initial estimate of the position and velocity direction could be derived using other navigation methods or by using radio pulsars themselves as is the case in the previous section. The estimate will help us to narrow down our choice of pulsars for which we would be performing the folding operation. This scenario will also allow us to reduce our velocity search space for navigation aboard celestial bodies as discussed in the following text.

When we know the coordinates and velocity direction of the body at the initial instant of navigation, we can determine which pulsars will be immediately available for tracking. This would in practice be made possible by the back end which would have access to a map of pulsars with respect to a reference system. In such a case, the back end would need to process the real time velocity direction and coordinates of the receiver and define its field of view allowing it to determine what pulsars the front end must search for. Once we know the pulsars for which folding should be performed, we are able to limit our pulsar search space to accommodate only the minimum number of pulsars that are needed for navigation at a particular time. In line to what has been derived before in the previous section, we would need a total of 12 pulsar sources.

As the back end has knowledge of the receiver location on a celestial body such as Earth, we can correct for the contribution of the celestial body's velocity in the total velocity of the receiver at a particular instant. Such a correction will help us reduce the velocity search space considerably. Based on our previous assumptions of the receiver velocity as in Section 4.3.3, the total velocity that a body would be traveling at would near 51 m/s.

The number of fixed point operations that are needed to be performed can again be calculated using Equation 5.1. Using the parameters as formulated above and those mentioned in Section 5.2.1, one deduces a value for Fi_{ops} to be $12.35 * (10^9)/s$. From Equation 5.3, at a sampling rate of 1 MHz, $MEM_{bandwidth}$ has a value of 40.1375 GB/s (both read and write). Also, from Equation 5.4; MEM_{total} is obtained to be 20.91 GB.

The summary of the computational requirements for the two scenarios is presented in the following table:

Scenario	Fi_{ops}/s	$MEM_{bandwidth}(GB/s)$	$MEM_{total}(GB)$
5.2.1	$5.25*10^{11}$	1706.25	853.125
5.2.2	$1.235*10^{10}$	40.1375	20.91

Table 5.2: Computational requirement summary

We can clearly observe that the memory bandwidth and size needed for this scenario are considerably less than those derived for the scenario in the previous section. The memory requirements have reduced by a factor of 40 approximately. Such memory specifications could be realistically met using multiple available DDR3 memory modules. We next present the architecture of the folding module as needed to be implemented on an FPGA and discuss the feasibility for the same.

5.3 FPGA Architecture and Hardware Requirements

The available on chip memory of the FPGA is much smaller than the needed memory storage capacity. We therefore need to make use of additional memory to the FPGA. The ML605 evaluation board provided by Xilinx makes use of a XC6VLX240T-1FFG1156 FPGA as its core. The external memory to the FPGA on the board is of the DDR3 type. We select this board for our feasibility study. An architecture depicting the internal and external memory is seen in Figure 5.1.

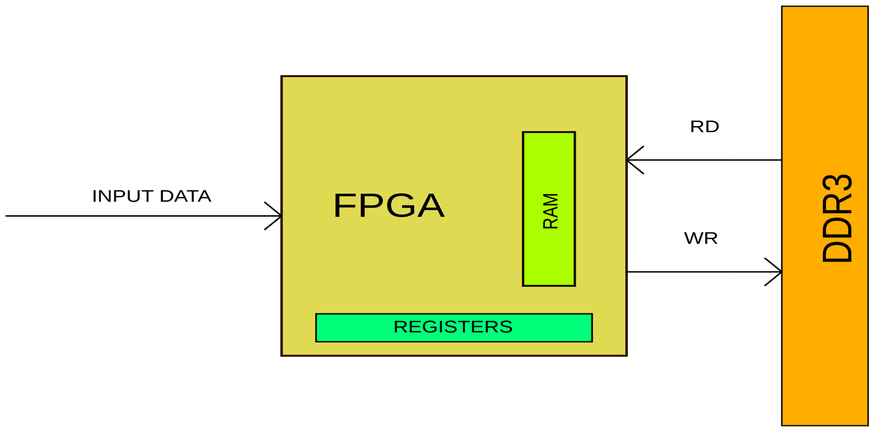


Figure 5.1: FPGA memories

Xilinx allows one to create DDR3 memory interface controllers within the FPGA logic fabric with the help of the Memory Interface Generator package [33]. The maximally supported theoretical memory bandwidth by the package is 6.4 GB/s and the maximum allowed memory size on the board is 2 GB [34]. The supported bandwidth is smaller than that which a DDR3 memory can provide. This is because, with each new generation of devices, the FPGA logic fabric timing improvement lags behind that of the DDR architecture's memory clock rate performance. This can be seen in Figure 5.2 [35].

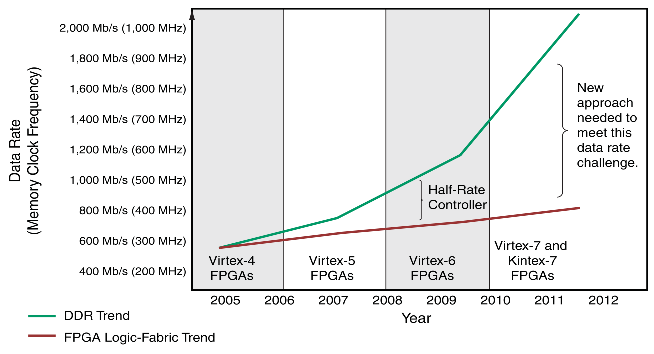


Figure 5.2: FPGA fabric vs DDR [35]

How many such boards do we need for our design? To be able to process a bandwidth nearing 40.1375 GB/s each read and write we first need to determine the number of parallel folds that a single board can accommodate, and then using the total number of channels that we need to fold; estimate the required number of boards. Here, a channel represents a period of a pulsar being folded in the FPGA. The estimation is undertaken in the following paragraphs.

The number of bits that are needed to represent a single folding sample has been derived to be 26 bits. In the practical implementation, data would be read and write as a 32 bit sample. For a single folding channel given an input data rate of 1 MHz, we will need a total of 32*1 M bits per second each for memory read and write to implement the folding process. For a single board, the total memory bandwidth is 6.4 GB (theoretical maximum). An efficient implementation would be to transfer one sample for a single channel at a time from the on chip RAM to the DDR3 and vice versa. The maximum theoretical number of folds that one board can perform using the external DDR3 memory is found to be

$$Channels_{perFPGA} = \frac{Board_{throughput}}{Channel_{throughput}} = \frac{6.4 * 10^9 * 8}{2 * 32 * 10^6} = 800\,. \tag{5.5}$$

The XC6VLX240T has a sufficient number of block RAM bits to allow storage of 800 samples. The on chip block RAM is 585 times bigger than our requirement. The data transactions within the FPGA (see Figure 5.3) would require additional memory, which can be accommodated using registers and the remaining block RAM. In Figure 5.3, the 'Memory Interface' is that generated by the Memory Interface Generator. The 'Folding Address Calculator' provides an indicator to the address of the accumulated samples in memory corresponding to the data sample to be folded at the next instant of time. Its structure would include a counter which will reset whenever the number of samples contained in a particular period are folded for, and an address mapper which will make use of a look up table to determine the location in the external RAM. 'Switch 1' directs the data from the memory to the respective channel on the FPGA. The data is transferred to channels one after the other in a cyclic manner. 'Switch 2' selects which address needs to be sent for data fetch from the DDR3. Here too channel requests are processed one after the other.

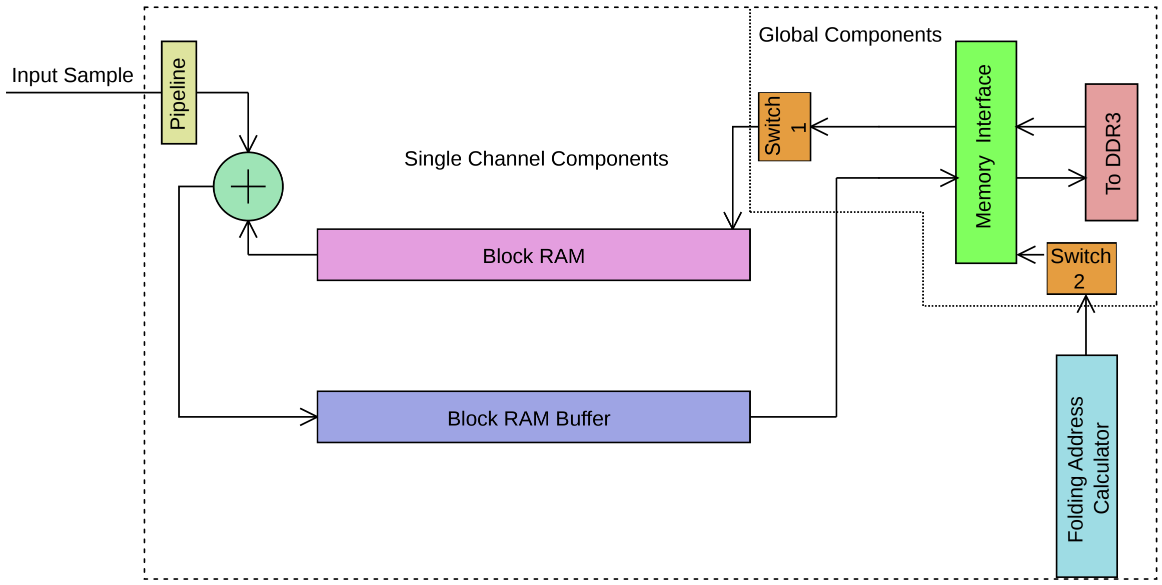


Figure 5.3: FPGA transactions

The number of channels that we need to fold for the scenario of Section 5.2.2 equal

12240. Hence, the theoretical minimum number of boards that will suffice our memory bandwidth requirement has a value of

$$\frac{Total_{channels}}{Channels_{perFPGA}} = \frac{12240}{800} \approx 16\,. \tag{5.6}$$

The memory size of 20.91 GB that is required for our implementation had been calculated assuming a sample to consist of 26 bits. Correcting this number to accommodate a 32 bit sample size results in a memory size of 25.73 GB. To accommodate this memory, we would need 13 boards as each can offer a 2 GB memory capacity. Since our hardware needs to have sufficient capacity both in terms of memory bandwidth and size, we require at least 16 boards to fulfill requirements in real time. The fixed point operation requirements would be distributed amongst the FPGAs leading to each computing $.8\, GFi_{ops}/s$, orders of magnitude smaller than what a single FPGA can maximally process.

5.4 Conclusion

It has been derived that for the case when a receiver does not know its initial position and velocity direction; the computational requirements needed to implement the folding subsystem are too large for an efficient implementation. We have simplified our usage scenario such that a receiver knows its initial coordinates and velocity direction. This allows us to reduce the needed hardware capacity by a factor of 40. Subsequently, we estimate that we would need hardware with specifications equaling those of at least 16 ML605 boards to demonstrate a prototype.

6 Evaluating Simulink based RTL generation

6.1 Model Based RTL Generation

From a Simulink description, one can now produce synthesizable VHDL or Verilog HDL codes towards an FPGA or an ASIC implementation. This is possible by using special Simulink blocks and design procedures that allow RTL generation. Such type of blocks are provided by the tool one uses to produce the RTL, and are integrated into the native Simulink environment provided by the Mathworks. An evaluation of this design methodology will serve as an aid in determining the best design approach of automatic RTL generation from high level descriptions, especially for our navigation system hardware. Even though an approach is best judged relative to others, we research the features and limitations of the methodology helping us to measure the efficacy of this for use in designing the navigation system.

Model based RTL generation presents several possible advantages as a design methodology for hardware. Such an approach may reduce the design cycle time for producing signal processing hardware. DSP modeling and simulations are commonly carried out in Matlab and Simulink, and hence it would assist if one can generate RTL code from Matlab/Simulink specifications directly. Simulink also supports features such as the Fixed Point Toolbox that allow easier modeling of hardware. In addition, Simulink offers visual analysis and debugging features that help in observing both the numeric results and the timing across the specified model. Usage of Simulink blocks for RTL generation would allow an architecture based high level synthesis approach which could result in a good translation into RTL.

However, as seen in this work, there are disadvantages when one makes use of such a design methodology. We find that the generated RTL is not optimized in the case of non trivial systems. We present the observed limitations in detail in Section 6.10.

There are several different tools that one can use to target RTL generation from the Simulink environment. They include: The Simulink HDL Coder provided by the Mathworks, Synphony HLS provided by Synopsys, and the System Generator by Xilinx. Each of these tools vary in their offered features. In the next section, a choice is made amongst these for producing RTL code towards an FPGA or ASIC implementation of the signal processing front end.

6.2 Tool Choices

The Xilinx System Generator is a tool that allows the generation of code from fixed point Simulink models using blocks contained in a Simulink toolbox that is provided by Xilinx. The code that is generated is only for Xilinx FPGAs. One cannot read the generated RTL code for some of the used blocks from the toolbox. This also does not allow optimization during synthesis. Since we seek an RTL code that should also find use in possibly an ASIC implementation of the navigation system, such a tool will be of no direct advantage. Hence we will be making a choice between the Simulink HDL Coder and Synphony HLS.

6.2.1 Synphony HLS and Simulink HDL Coder

The hardware definitions in Simulink using either of the tools make use of two different block types. The first are IP blocks that are provided in the tool's toolbox. The second are user coded Matlab blocks that are integrated into Simulink. The IP blocks comprise of DSP blocks, memories, arithmetic operators, communication blocks, logic blocks, signal sources, and others. Simulink HDL Coder offers a lot more IP blocks, approximately 170 [36] as compared to 80 by Synphony HLS [37]. As an example, Synphony HLS does not provide a dual port memory block. The HDL Coder also offers more features for converting user coded Matlab to RTL code in Simulink. The HDL Coder supports 270 functions in the user coded Matlab block, significantly more than the 50 offered by Synphony HLS. As an example, Synphony HLS does not support user coded division in many cases [38]. Based on feedback from a user, the HDL Coder is said to generate a more readable and intelligent code than Synphony HLS as well. Hence for our use, we have adopted the Simulink HDL Coder (version 2.1).

6.3 Design Flow

The design flow of HDL generation using Simulink HDL Coder can be understood with the help of the following block diagram:

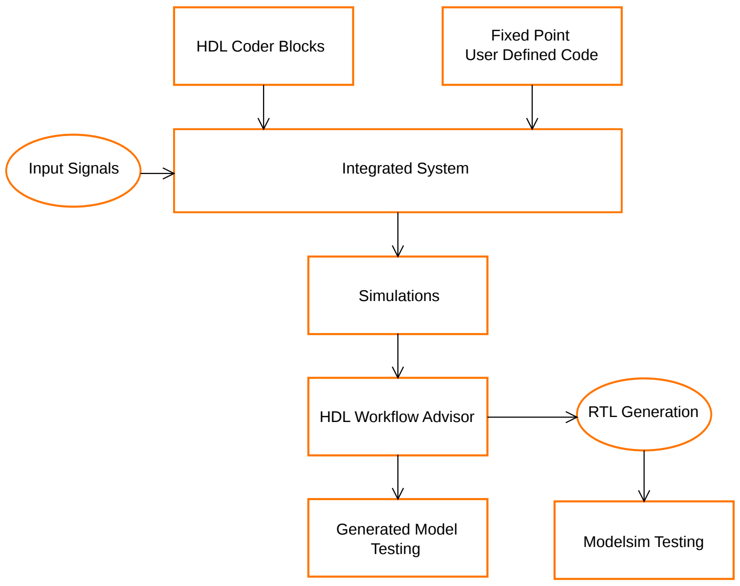


Figure 6.1: HDL Coder design flow

The user first partially describes the hardware in Simulink using only those blocks that are supported by the HDL Coder. For modeling components that are not available in the Coder library, one makes use of the user defined Matlab code block. The language to program the Matlab code block is the Embedded Matlab language, differing from the standard M code. There are some limitations when modeling using the Embedded Matlab language as compared to using the standalone M language. We investigate the performance of RTL generation using both the IP and user coded block types. Also, the entire specification of IP and user coded blocks should be in fixed point format to allow RTL generation.

After the system has been created, simulations are undertaken to check if the numeric and timing results are as required. The next step is running the HDL Workflow Advisor which helps fixing the model settings to those as required for generating HDL. Examples of settings include the Simulink solver type, the simulation step size, and the fixed point specifications. The Advisor also provides different high level design choices for RTL generation which include the generation of a clock enable, the clock sensitivity of registers, generation of a test bench, and others.

Once all the steps in the Advisor have been successfully executed, one obtains an RTL description of the Simulink model along with an optional RTL test bench, as well as a 'Generated model'. The RTL test bench provides as input the same input signal values that one uses during simulation. The Generated model mimics the specified hardware in RTL using a Simulink model which again can be verified with respect to the timing and numeric results on simulation. The RTL test bench can be executed in Modelsim to test for the functional correctness of the RTL design.

6.4 HDL Coder Features

6.4.1 Integrating legacy HDL code

One can integrate a manual HDL coded block in a Simulink model and still be able to perform simulations and generate HDL for the system. To allow this feature, one requires an additional toolbox called the EDA simulator link also provided by the Mathworks. This feature offers a designer to make use of HDL generated/coded by other means in place of those blocks for which Simulink is not able to produce efficient RTL.

6.4.2 Integrated synthesis

It is possible to integrate the Xilinx ISE and Altera Quartus environments into the HDL Workflow Advisor. This allows one to synthesize as well as perform placement and routing for an FPGA or a few OEM boards within the Simulink environment. This also allows one to identify within the Simulink model, the three most timing critical paths. This feature should help in reducing the design cycle time.

6.4.3 Traceability

This is a feature which allows one to view the corresponding generated HDL code for any particular block in the design. One can jump from a block to its corresponding HDL and vice versa, allowing better design analysis and user feedback. The RTL code that is generated by HDL Coder is found to be very much readable.

6.4.4 Resource utilization report

The HDL coder presents to the designer, a hardware resource utilization report. The report is produced after generation of the RTL code. The report includes the number of multipliers, adders/subtracters, registers, RAMs and multiplexers used in the hardware design. The report also states the specifications for each of the above used component type.

6.5 Optimization Features

We are offered three architecture level optimizations namely: distributed pipelining, resource sharing, and resource streaming. We present the concepts of each in the following text.

6.5.1 Resource streaming

This feature allows one to reduce the hardware required for vector operations. Given a vector or an array that is being operated on in Simulink, by default all array elements are processed in parallel in the same clock cycle. One can specify a streaming factor which results in lesser hardware resources operating at a faster rate.

6.5.2 Resource sharing

Across a subsystem in the Simulink model, one can apply resource sharing for reducing the hardware area. Simulink HDL coder allows for sharing amongst: multipliers, gain blocks, and atomic subsystems. Each of the component that needs to be shared should have identical copies with respect to component specifications within the same subsystem.

6.5.3 Distributed pipelining

The Coder offers the possibility to the designer of automatically inserting pipeline registers along the critical path. It allows higher throughputs at the cost of an increased latency. Distributed pipelining is undertaken by selecting a number of pipeline registers at the output of a subsystem and setting the distributed pipelining option as on in the subsystem properties. At the generation of the RTL code, one can observe the distributed pipeline in the Generated model. It is also suggested to run retiming during synthesis for further timing optimizations in our design.

6.5.4 Delay balancing

Using any of the above three optimization techniques can lead to there being an introduction of delays along the critical path in the model. This can lead to unwanted numeric errors in the output results. To solve this, the HDL coder uses the technique of delay balancing wherein matching delays are inserted along parallel paths to those where optimization delays were added. One can make use of the 'Verification Model' to check for consistencies between the original model outputs and the delay balanced outputs.

6.6 Folding Algorithm Design

We implement the folding module of our navigation system in Simulink. This will help us in our goal of rating the methodology of automatic RTL generation from Simulink models.

6.6.1 Simulink model

The top level Simulink model of the generic (target independent) folding module is presented in Figure 6.2. It comprises of IP blocks and a user coded block. The architecture is based on the available IP blocks, and is chosen over other possibilities for the reason of reducing hardware costs. The model is designed to fold at any period specified by the back end in real time. There are two inputs to the block. The first (*Data*) is the real time data sample input from the output of the dedispersion block. The input dynamic range is represented by the unsigned data type with 4 digits to the right and 1 digit to the left of the decimal point. The second (*Sample_number*) is the number of samples for which folding is to be performed. This number is fed by the back end and denotes the number of samples contained in the period for which we are folding. We represent this value using the uint32 data type so as to allow large values of *Sample_number*. The data is stored and folded in an accumulator. Here, we have assigned the accumulator to consist of 100 samples. Each sample has B number of bits, see Section 5.2.1. In the depicted model for simplification, only one pulsar period is being folded for. We have also simulated for parallel folds at different periods and accumulator sizes, see Section 6.8.

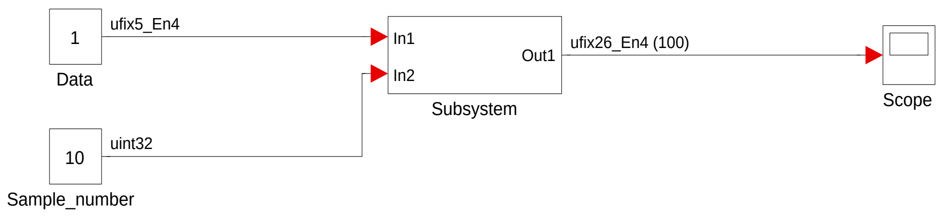


Figure 6.2: Top level model

Going down a level further as shown in Figure 6.3, we see the model to be comprised of control elements (namely a counter, a relational operator and a logical operator) and a *User_coded* subsystem. The user coded subsystem contains a block that encapsulates code written in the Embedded Matlab language. The user coded block adds an input sample to the corresponding phased samples from the previous periods in the accumulator. The control elements provide a count to the user coded block which it uses as a memory index to the accumulator as well as uses to check the completion of folding for a single period. After folding one period, we send the result of the accumulator as an output of the model. The process continues resulting in outputs folded for several periods. In the RTL that is generated, there is by default a reset port which can be used to refresh the contents of the accumulator as and when the remaining hardware system may desire.

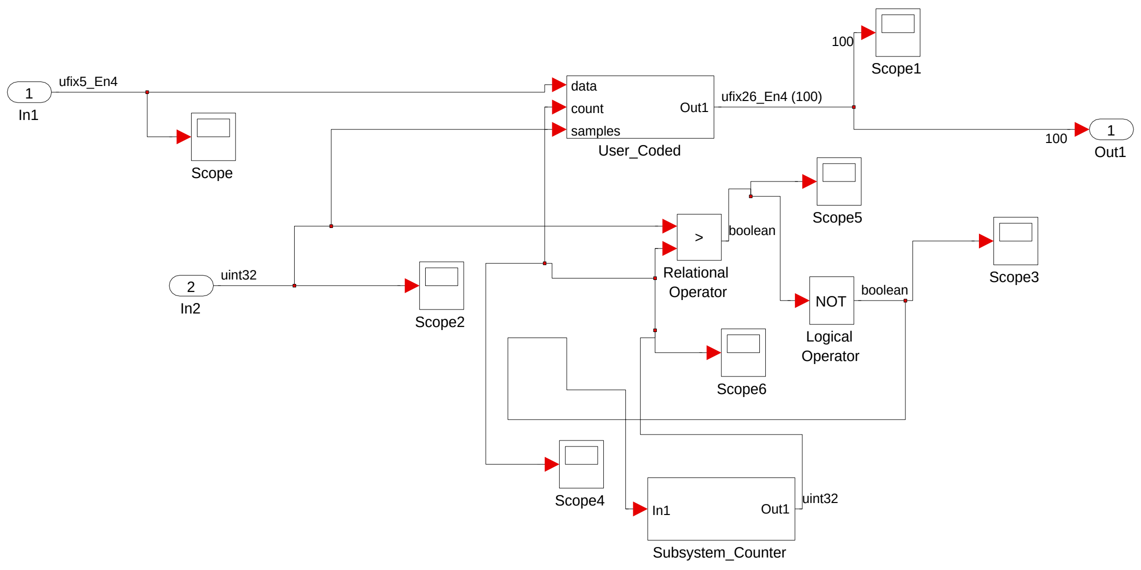


Figure 6.3: Subsystem model

The *Subsystem_counter* consists of a HDL Coder library element; a free running counter with the input being a reset port. The logic elements (relational and logical operators) check if the number of samples contained in a pulsar signal period is greater than the number of samples that have participated in the folding process for a single period. When this cedes to be true, the counter is reset to its initial value, and now the index of the accumulator in the user coded block is set to the starting sample again. The user code is presented in Figure 6.4.

```
function y= fcn(data,count,samples)

hdl_fm = fimath(...                        %Fixed point specifications
'RoundMode', 'floor',...
'OverflowMode', 'saturate',...
'ProductMode', 'FullPrecision', 'ProductWordLength', 32,...
'SumMode', 'FullPrecision', 'SumWordLength', 32,...
'CastBeforeSum', true);

persistent price;                          %Memory variable ~ Accumulator
if(isempty(price))                         %Initialization
   price = fi(zeros(100,1), 0, 26, 4, hdl_fm);
end

temp = price(count,1);                     %Temporary storage

%Adding a new sample to corresponding previous
%samples in the accumulator
price(count,1)= data+temp;

if (count)==samples                        %Folding complete?

    y=price;                               %Send non zero output

else
 y=fi(zeros(100,1),0,26,4,hdl_fm);         %Output remains zero
end
```

Figure 6.4: User code

6.7 Simulation Results and Synthesizability

The Simulink model is run and we have observed the results using the Scope block. One can view the folding operation being performed as desired. Upon completing steps in the Workflow Advisor, the Generated model also produces the same result.

We also simulate the RTL test bench generated by the HDL coder in Modelsim. Here too we observe the RTL simulations to depict the folding operation. Finally, a synthesis of the RTL is successfully undertaken using Synplify Pro, testifying that HDL Coder produces a synthesizable RTL code. In Figure 6.7, one can see the simulation results from Modelsim after three folds of the input signal.

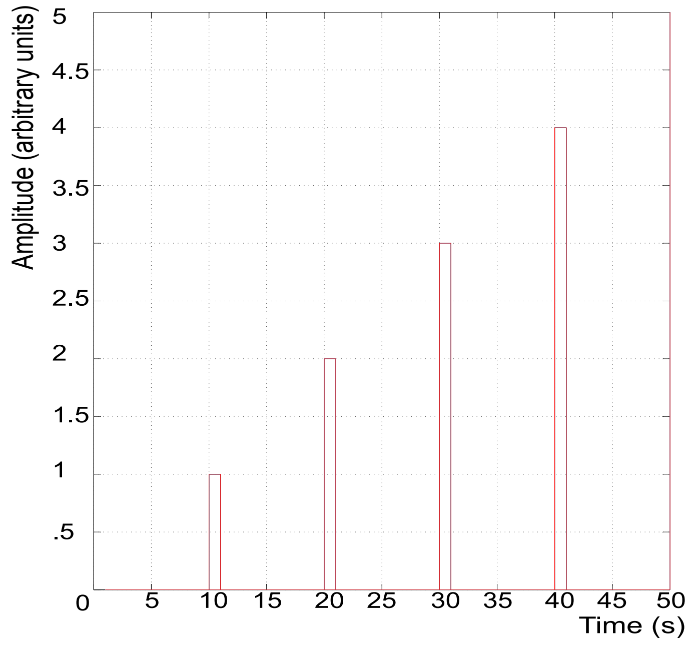


Figure 6.5: Simulink model simulation result

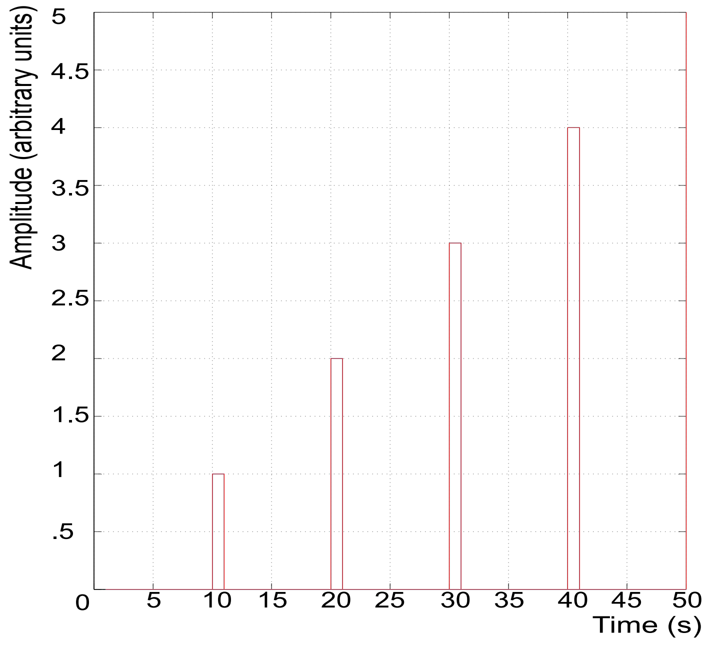
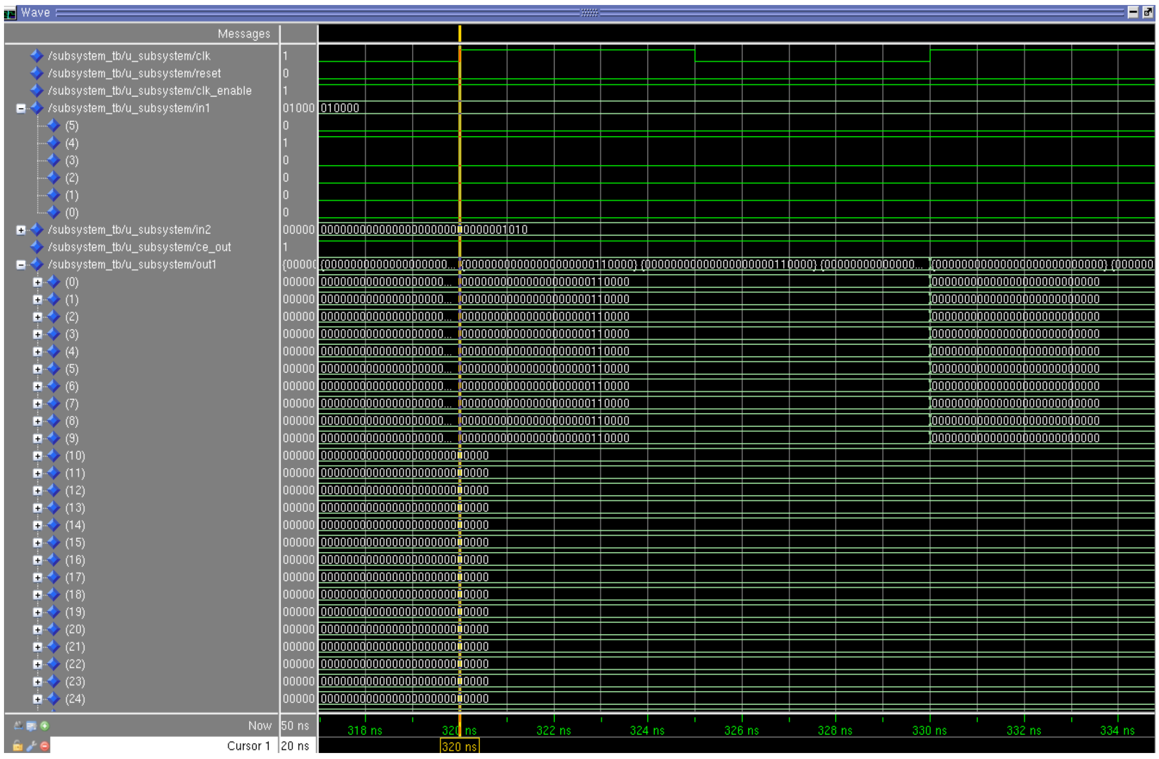


Figure 6.6: Generated model simulation result



```
Wave
                                  Messages
/subsystem_tb/u_subsystem/clk          1
/subsystem_tb/u_subsystem/reset        0
/subsystem_tb/u_subsystem/clk_enable   1
/subsystem_tb/u_subsystem/in1          01000
   (5)                                 0
   (4)                                 1
   (3)                                 0
   (2)                                 0
   (1)                                 0
   (0)                                 0
/subsystem_tb/u_subsystem/in2          00000
/subsystem_tb/u_subsystem/ce_out       1
/subsystem_tb/u_subsystem/out1         {00000
   (0)                                 00000
   (1)                                 00000
   (2)                                 00000
   (3)                                 00000
   (4)                                 00000
   (5)                                 00000
   (6)                                 00000
   (7)                                 00000
   (8)                                 00000
   (9)                                 00000
   (10)                                00000
   (11)                                00000
   (12)                                00000
   (13)                                00000
   (14)                                00000
   (15)                                00000
   (16)                                00000
   (17)                                00000
   (18)                                00000
   (19)                                00000
   (20)                                00000
   (21)                                00000
   (22)                                00000
   (23)                                00000
   (24)                                00000
                                  Now  50 ns
                             Cursor 1  20 ns
```

Figure 6.7: Modelsim result

6.8 Scalability

One can by making use of custom scripts automate the process of scaling Simulink models. Simulink allows one to add additional blocks to a model, make connections between blocks as well as change block properties all by using commands in the Matlab prompt. We have scaled our model with respect to the number of parallel channels that are being folded simultaneously, the bit width of the folded samples as well as the accumulator size in the user coded block. A design undertaking several folds in parallel is part of our needed real world hardware. Also, based on the signal sampling rate and the periods that the back end decides to use for navigation, we can have a group of varied folding engines whose accumulators differ in their range of supported folding periods. This will help us to reduce real time memory requirements. An example of a scaled model that performs multiple parallel folds can be seen in Figure 6.8. Upon calculating the number of channels that we would be folding for and the required memory size for each, we can appropriately scale our model.

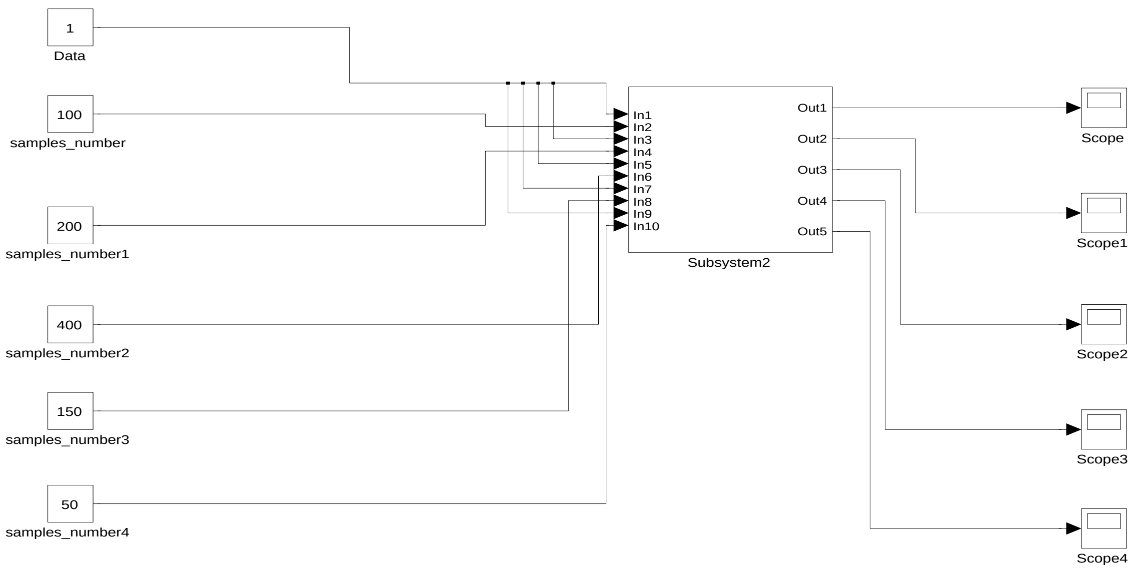


Figure 6.8: Multiple parallel folds

6.9 IP Blocks for Dedispersion

The HDL coder provides the filter and FFT block IPs for RTL generation. These are of benefit when we want to design the dedispersion block using the Simulink HDL Coder. The FFT IP generates a radix-2 based minimum resource implementation. There is also a provision of inserting i/o pipelines. The filter block offers the possibility of a fully parallel or a serialized implementation [36]. The performance of the IPs can be measured against a custom HDL code or that generated by other methodologies.

6.10 HDL Coder Limitations

Thus far we have seen that the HDL Coder provides a convenient platform to design and generate synthesizable RTL. However, is the generated RTL suitable for efficient FPGA or ASIC target solutions? Certainly not without the partial list of limitations enumerated below.

1. The user coded block is always processed in one clock cycle. One cannot make use of the resource sharing and streaming optimizations on such a block. This can generally lead to a hardware description that makes use of a large number of resources. The user would need to manually partition the hardware description.

2. There is no possibility to perform distributed pipelining for a user coded block if it makes use of a memory (persistent variable). This limitation also affects our RTL design.

3. The Scope block graph generations proceed very slowly when large vectors are used in the design. In our model, when we set the accumulator size to be 1 Mega samples, the time approaches several hours. This causes our design cycle time to have an additional overhead. An even more time consuming step is generation of the RTL test bench when large vectors are involved. It takes several days to generate test benches for designs making use of large arrays.

4. The architectural design space of a hardware system is constrained by the available IP cores. This may reduce the ability of the Coder to model a complete system, and generate an RTL for it.

5. It is difficult to realize complex blocks efficiently if the constituent sub blocks are not already supported by the HDL Coder for RTL conversion. For our navigation system, we most definitely need to implement correlation in hardware. However, even though correlation is a standard signal processing algorithm, there is no correlation block supported by HDL Coder. It would be inefficient to realize correlation using the user coded block, and hence we need to make use of another design methodology.

6. The resource streaming and sharing optimizations do not work when a feedback loop is present in the design. In our design, the control elements and the HDL counter form a feedback loop inhibiting any possibility of such optimizations.

7. Even though the HDL counter block is provided by the HDL Coder, it does not support delay balancing preventing any architecture optimizations on this block. This indicates that even when using HDL Coder supported hardware blocks, we can face limitations on pipelining and resource usage optimizations.

8. Distributed pipelining across a system is not possible if it contains any block that is clock enabled.

9. A major limitation of using HDL Coder for RTL generation is the way memories are mapped from Simulink models to RTL. The memories in the user code are always mapped to registers in RTL. There is no provision for them to be mapped to RAM blocks. A design could be made that makes use of the available HDL Coder RAM blocks instead of a memory in the user coded block. However, the sizes of the RAM blocks that are available allow only an address width of 16 bits. This prevents us from making use of a RAM block to model a large accumulator, which would be subsequently split into smaller RAM blocks by a synthesis tool when targeting an FPGA or could be replaced by a single IP when targeting an ASIC or external FPGA memory. Hence, systems that involve a large memory are not suited for implementation using the HDL Coder.

6.11 Conclusion

We have seen that Simulink HDL Coder provides a user friendly platform for system simulation and RTL generation. Following this methodology, one reduces the time required to produce a synthesizable RTL. The HDL Coder also allows for easy verification of hardware descriptions. However, when it comes to the generation of resource optimized RTL and memory block extractions from the high level descriptions, the tool's current version lacks features limiting its possibility of being the main design tool for large hardware systems.

Summary and Future Work 7

7.1 Summary

In this work, we have progressed in the conceptual understanding and design of a navigation system that makes use of radio pulsar signals. This work addressed the detection of pulsar signals on a moving receiver as well as estimating the velocity with which the receiver is moving with respect to pulsars. We have identified the usage of such a system to be on ships and rovers on planets and on spacecrafts. Knowledge of the techniques of radio pulsar astronomical instrumentation has allowed us to select key algorithms that would aid in navigation. These include coherent dedispersion in frequency domain, folding in time domain, and correlation. Our choices have been made after considering that radio pulsar signals have SNRs nearing -70 dB when received by antennas of size around 10 m^2 on Earth. We also researched on system parameters such as the frequency of pulsar observations that will be most suitable for navigation. After narrowing down on design choices, we constructed the system's modular architecture that allows us to navigate using pulsar signals that were dispersed and Doppler shifted when being received at the input antenna.

We then calculated the SNR enhancement that the folding subsystem provides for a non stationary receiver with respect to the pulsar. This was undertaken to determine if we need to implement the technique of dedispersion, and if so which of the two dedispersion techniques should we make use of. The analysis was first carried out for a spacecraft and then for a moving receiver on a planet or some other similar celestial body. We found that for a practical implementation, folding provides an SNR improvement of about 20 dB. We then discussed that correlation of folded signals with pulsar templates would need an SNR in the range of 0 dB to 25 dB. Subsequently, we determined the enhancement dedispersion must provide and proposed the bandwidth needed to be dedispersed coherently.

Following the above was a trial of using Simulink based models for RTL generation. This was in line of our goal to find out the feasibility of using Simulink for generating synthesizable RTL for our navigation system modules. We made a choice amongst different tools that allow such a design methodology and selected Simulink HDL Coder; which allows the generation of target independent synthesizable RTL. Based on our design of the folding subsystem and subsequent experimentation, we conclude that the current version of the HDL Coder lacks strong enough features that should otherwise result in an optimized RTL for an FPGA or an ASIC implementation.

Finally, we determined the computational requirements of the folding subsystem for it to be implemented in real time. We observed that for the case when the initial coordinates and velocity direction of the rover or ship with respect to the celestial body on which it is moving are unknown; the hardware requirements are too large to allow a practically efficient implementation. We thereby limit our design to the scenario where both the initial position on the celestial body, and the velocity direction are known. Based on the required fixed point operations per second, memory bandwidth and memory size; we have calculated the number of ML605 evaluation boards that we would need to demonstrate a complete prototype to be at least 16.

7.2 Future Work

1. When targeting navigation on Earth, a technique towards Radio Frequency Interference Mitigation may be implemented as in [19]. This could allow an abatement of noise generated by man made radio sources, resulting in an enhancement of the received pulsar SNR.

2. To test the signal processing front end implementations, it is desirable to have a large number of coherently dispersed pulsar data. We can create a program that generates analog/digitized artificial pulsar signals.

3. We need to study the effect of relativity on our navigation system. A pulsar signal as would be seen and recorded in our frame of reference could vary from that which is observed in a region of different gravitational potential. Also, the used clocks for navigation could emit time more slowly when placed on board a spacecraft moving at a high velocity as well as in the presence of a high gravitational field.

4. The correlation block needs future work to determine the exact value of the correlation coefficient that would allow the desired accuracy when navigating. Also, we need to determine which of the two: time or frequency domain processing would be more advantageous.

5. We need to devise a computationally feasible signal processing back end. The back end must also be able to generate pulsar profiles, possibly including deviations caused by relativity, as would be seen at different receiver velocities. After construction of the back end algorithms, the necessary velocity resolution value by the back end can be used for a more appropriate analysis of the SNR enhancement by folding.

6. The methodology of C to RTL generation using the Catapult C tool could be tested. This could be an alternative for producing optimized RTL code from high level specifications in a short time. Based on its documentation [39], it appears to offer better memory extractions than HDL Coder version 2.1. We can compare the performance of the available IP provided by Catapult C and Simulink HDL Coder towards implementing the dedispersion block. Also, we can check the improvements offered by the newer versions of the HDL Coder and reassess its usability.

7. Coherent dedispersion and folding should be implemented on FPGAs.

Pulsar SNRs

A

The table below presents the SNR values of a few pulsars [15].

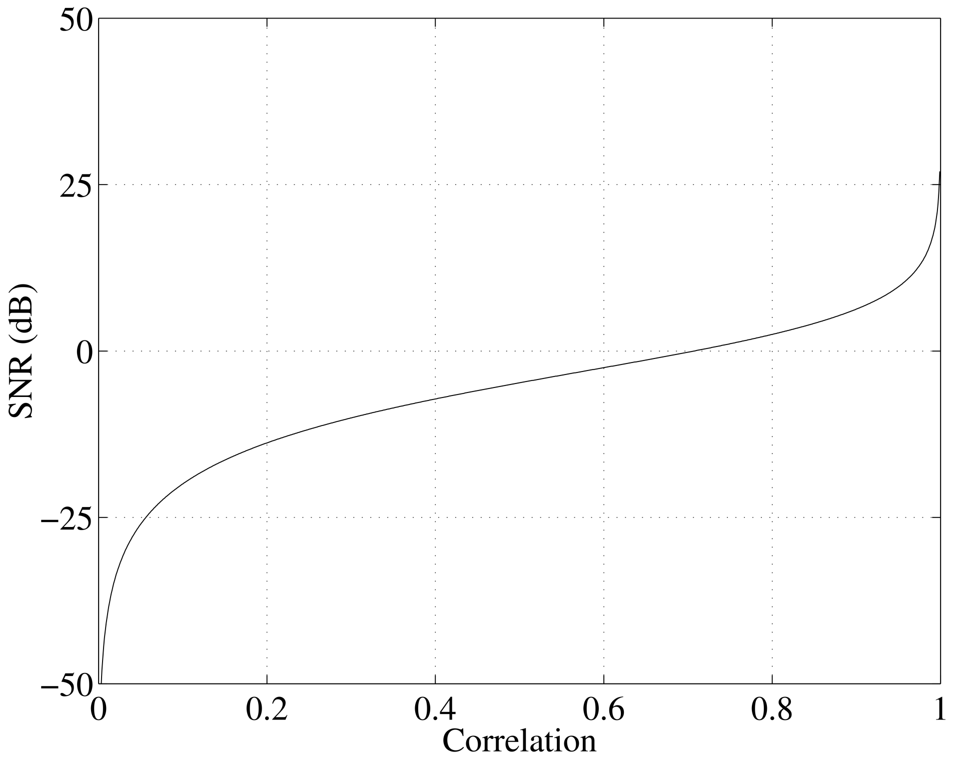
Name	l_E[a]	b_E[a]	l[b]	b[b]	α[c]	δ[c]	DM[d]	Spectral[e]	$S_{\nu,T}$[f]	T_{gal}[g]	T_{sys}[h]	SNR[i]	Q[j]
	(deg)	(deg)	(deg)	(deg)	(h)	(deg)	(cm^{-3} pc)	Index	(mJy)	(K)	(K)	(dB)	(dB)
B1937+21	301.82	42.30	57.51	−0.29	19.661	21.58	71.04	−2.2	33.1	10.8	43.3	−55.6	11.8
B0329+54	65.19	34.26	145.00	−1.22	3.550	54.58	26.83	−1.6	347.4	9.2	41.9	−45.2	4.5
B1642−03	250.19	18.86	14.11	26.06	16.751	−3.30	35.73	−2.3	46.1	6.4	39.4	−53.7	1.3
B0950+08	147.71	−4.62	228.91	43.70	9.886	7.93	2.96	−1.2	127.7	2.6	35.9	−48.9	−0.7
B0740−28	125.33	−48.71	243.77	−2.44	7.714	−28.38	73.76	−2.4	33.4	5.2	38.3	−55.0	−2.3
B1929+10	297.05	32.29	47.38	−3.88	19.537	10.99	3.18	−1.7	63.8	15.0	47.1	−53.1	−2.9
B1556−44	246.80	−23.57	334.54	6.37	15.995	−44.65	56.10	−0.8	52.5	24.9	56.0	−54.7	−3.8
B1451−68	246.26	−49.00	313.87	−8.54	14.933	−68.73	8.60	−1.2	118.9	10.7	43.2	−50.0	−3.8
B1449−64	243.26	−45.02	315.73	−4.43	14.892	−64.22	71.07	−2.2	29.7	16.0	48.0	−56.5	−4.1
B0835−41	150.63	−57.04	260.90	−0.34	8.623	−41.59	147.29	−2.0	31.4	6.7	39.6	−55.4	−4.1
B2020+28	317.95	46.59	68.86	−4.67	20.377	28.91	24.64	−0.5	44.9	11.7	44.1	−54.4	−4.2
J1617−5055	251.79	−29.09	332.50	−0.28	16.291	−50.92	467.00	−1.7	65.6	37.1	67.0	−54.5	−5.3
J1730−2304	263.14	0.19	3.14	6.02	17.506	−23.08	9.61	−2.0	7.2	23.2	54.5	−63.3	−6.4
B1821−24	275.57	−1.55	7.80	−5.58	18.409	−24.87	119.86	−4.3	0.8	34.8	64.9	−73.7	−7.7
J2322+2057	0.04	22.88	96.52	−37.31	23.373	20.95	13.37	−1.7	0.6	3.5	36.7	−72.4	−8.5
B2016+28	316.46	46.69	68.10	−3.98	20.301	28.67	14.17	−1.9	56.4	11.3	43.8	−53.3	−4.4
B1749−28	268.45	−4.68	1.54	−0.96	17.883	−28.11	50.37	−3.3	54.3	49.6	78.2	−56.0	−4.5
B0823+26	122.60	7.24	196.96	31.74	8.448	26.62	19.45	−1.6	17.1	2.8	36.1	−57.7	−4.6
B1240−64	225.40	−52.66	302.05	−1.53	12.721	−64.39	297.25	−1.7	23.1	13.2	45.5	−57.4	−4.6
B1933+16	299.33	37.31	52.44	−2.09	19.597	16.28	158.52	−1.4	67.2	13.2	45.5	−52.7	−4.7
B2310+42	10.60	43.12	104.41	−16.42	23.219	42.89	17.28	−1.4	24.2	5.6	38.6	−56.5	−5.8
B1133+16	168.15	12.16	241.90	69.20	11.601	15.85	4.86	−1.7	56.0	2.9	36.2	−52.5	−6.0
B2021+51	337.86	67.13	87.86	8.38	20.381	51.91	22.65	−0.8	35.8	10.1	42.7	−55.2	−6.1
B1557−50	248.66	−29.46	330.69	1.63	16.015	−50.74	260.56	−1.7	30.1	35.0	65.1	−57.8	−6.2
B1821−19	275.66	3.56	12.28	−3.11	18.400	−19.76	224.65	−2.1	10.0	31.7	62.1	−62.3	−6.4
B1356−60	233.08	−44.59	311.24	1.13	14.000	−60.64	293.71	−2.1	15.4	20.9	52.4	−59.8	−6.6
B1323−62	229.57	−48.10	307.07	0.21	13.455	−62.38	318.80	−1.7	28.4	19.5	51.1	−57.0	−7.1
B0736−40	130.28	−60.75	254.19	−9.19	7.642	−40.71	160.80	−0.7	100.9	5.2	38.3	−50.2	−7.5
B1713−40	261.42	−17.78	346.82	−1.73	17.289	−40.91	308.50	1.0	38.3	32.2	62.6	−56.6	−8.2
B1804−08	271.95	14.63	20.06	5.59	18.127	−8.80	112.38	−1.2	22.2	22.2	53.6	−58.3	−8.5
B1323−58	227.02	−45.27	307.50	3.57	13.450	−58.99	287.30	−2.0	19.3	16.2	48.2	−58.4	−8.7
B1508+55	188.03	67.22	91.33	52.29	15.157	55.53	19.61	−2.1	16.3	3.3	36.6	−57.9	−8.7
B1911−04	289.34	17.53	31.31	−7.12	19.232	−4.68	89.39	−2.6	10.6	22.2	53.6	−61.5	−8.7
B0136+57	48.56	43.80	129.22	−4.04	1.655	58.24	73.78	−1.4	7.5	9.5	42.1	−61.9	−8.9
B2217+47	3.29	52.54	98.39	−7.60	22.330	47.91	43.52	−2.9	7.9	7.4	40.3	−61.5	−9.1
B1046−58	201.08	−57.84	287.43	0.58	10.803	−58.53	129.10	−1.7	11.5	12.5	44.8	−60.3	−9.2
B1055−52	195.47	−52.39	285.98	6.65	10.966	−52.45	30.10	−1.7	16.8	6.2	39.2	−58.1	−9.3
B1221−63	222.53	−53.82	299.98	−1.42	12.406	−64.13	97.47	−2.0	7.7	11.6	44.0	−62.0	−9.4
B1702−19	257.14	3.72	3.19	13.03	17.093	−19.11	22.91	−1.0	11.3	9.9	42.5	−60.2	−9.5
B1953+50	327.05	68.80	84.79	11.55	19.922	51.00	31.97	6.6	8.6	41.3	41.3	−62.4	−9.7
B0450+55	79.04	32.91	152.62	7.55	4.902	55.73	14.49	19.5	8	40.8	40.8	−57.6	−9.7
B1727−47	264.76	−24.43	342.57	−7.67	17.528	−47.74	123.33	−2.2	25.2	17.7	49.5	−57.4	−9.9
B0540+23	86.14	0.10	184.36	−3.32	5.719	23.48	77.71	12.3	7.5	40.3	40.3	−59.6	−10.0
B1600−49	248.77	−27.80	332.15	2.44	16.073	−49.17	140.80	−1.7	9.6	34.0	64.2	−62.7	−10.1
B1818−04	275.50	18.88	25.46	4.73	18.348	−4.46	84.44	−2.6	14.6	25.3	56.4	−60.3	−10.4
B1358−63	235.86	−47.38	310.57	−2.14	14.031	−63.96	98.00	−1.4	9.8	19.5	51.1	−61.6	−10.5
B1426−66	241.53	−48.01	312.65	−5.40	14.511	−66.38	65.30	16.9	14.8	46.9	46.9	−58.9	−10.7
B1742−30	266.95	−7.27	358.55	−0.96	17.766	−30.67	88.37	−1.3	20.1	47.9	76.7	−60.3	−11.2
B1607−52	250.86	−30.50	330.92	−0.48	16.184	−52.16	127.57	2.1	36.5	66.4	66.4	−69.4	−11.6
B1719−37	262.29	−14.00	350.49	−0.51	17.383	−37.20	99.50	−1.6	5.6	37.2	67.1	−65.3	−11.7

[a] Ecliptic longitude and latitude.
[b] Galactic longitude and latitude.
[c] Right ascension and declination $J2000$.
[d] Dispersion measure.
[e] Between 0.4 and 1.4 GHz.
[f] At 1 GHz, average over the pulsar period
[g] Assuming $\eta_M = 0.9$.
[h] Assuming no contribution from the Sun.
[i] Assuming $A_e = 10$ m^2.
[j] Quality factor per 10^9 samples, referred to an accuracy of 10^6 m.

Correlation versus SNR

B

The plot of correlation versus SNR. The correlation is between a signal and its noise corrupted version [32].



Bibliography

[1] S. I. Sheikh and D. J. Pines, "Recursive estimation of spacecraft position and velocity using X-ray pulsar time of arrival measurements," in *Navigation Journal of the Institute of Navigation*, vol. 53, pp. 149–166, 2006.

[2] A. A. Kestila, "Deep space navigation system using radio puslars: Back-end," Master's thesis, Delft University of Technology, The Netherlands, August 2009.

[3] R. Resnick, D. Halliday, and J. Walker, *Fundamentals of Physics*. Wiley Publications, 8 ed., 2007.

[4] S. Engelen, "Deep space navigation system using radio puslars: Front-end," Master's thesis, Delft University of Technology, August 2009.

[5] D. Lorimer and M. Kramer, *Handbook of Pulsar Astronomy*. Cambridge University Press, 2 ed., 1998.

[6] J. G. Hartnett and A. N. Luiten, "Colloquium: Comparison of astrophysical and terrestrial frequency standards," in *Reviews of Modern Physics*, vol. 83, January - March 2011.

[7] R. N. Manchester, "vela.avi." `www.atnf.csiro.au/research/pulsar/audio`, Accessed September 2011.

[8] P. Graven *et al.*, "XNAV for deep space navigation," in *Annual AAS Guidance and Control Conference*, vol. 31, 2008.

[9] "CSIRO." `http://outreach.atnf.csiro.au/education/everyone/pulsars/`, Accessed July 2011.

[10] "NRAO." `http://www.nrao.edu/pr/2006/mspulsar/`, Accessed July 2011.

[11] "NASA." `http://deepspace.jpl.nasa.gov/dsn/features/wherearethey.html`, Accessed July 2011.

[12] P. S. Ray *et al.*, "Deep space navigation using celestial X-ray sources," in *2008 National Technical Meeting of The Institute of Navigation*, (San Diego, CA), pp. 101–109, January 2008.

[13] S.A.Butman and T.J.Chester, "Navigation using X-ray pulsars," Tech. Rep. N81-27129, NASA, 1981.

[14] J. Dong, "Pulsar navigation in the solar system," in *Arxiv.org*, February 2011.

[15] J. Sala *et al.*, "Feasibility study for a spacecraft navigation system relying on pulsar timing information," Tech. Rep. 03/4202, ARIADNA Study, June 2004.

[16] G. Hobbs and R. Manchester, “TEMPO2, a new pulsar-timing package - i. an overview,” in *Monthly Notices of the Royal Astronomical Society*, vol. 369, pp. 655–672, June 2006.

[17] W. Straten and M. Bailes, “DSPSR: Digital signal processing software for pulsar astronomy,” in *Publications of the Astronomical Society of Australia*, vol. 28, January 2011.

[18] D. Bhattacharya *et al.*, “Many faces of neutron stars,” in *NATO ASI*, Kluwer Academic Publishers, pp. 103–128, 1998.

[19] D. Allal *et al.*, “RFI mitigation in the context of pulsar coherent de-dispersion at the Nancay radio astronomical observatory,” in *European Signal Processing Conference*, 2009.

[20] T. H. Hankins and B. J. Rickett, “Pulsar signal processing,” in *Methods in Computational Physics*, vol. 14, 1975.

[21] A. G. Lyne, *Pulsar Astronomy.* Cambridge University Press, 2 ed., 1998.

[22] P. J. Buist *et al.*, “Overview of pulsar navigation: Past, present and future trends,” in *Navigation: Journal of The Institute of Navigation*, vol. 58, pp. 153–164, Summer 2011.

[23] K. O’Neil, “Current and future instrumentation on the 100m Green Bank Telescope,” in *Proceedings of the ISKAF2010 Science Meeting*, (Assen, The Netherlands), June 2010.

[24] S. M. Ransom *et al.*, “GUPPI: Green Bank Ultimate Pulsar Processing Instrument,” in *AAS*, vol. 214, p. 605.18, 2009.

[25] D.R.Lorimer *et al.*, “A 1400-Mhz pilot search for young pulsars,” in *Astronomy and Astrophysics*, vol. 358, pp. 169–176, 2000.

[26] Analog Devices, *AD9467: 16-Bit, 200 MSPS/250 MSPS Analog-to-Digital Converter*, March 2011.

[27] Z. Arzoumanian *et al.*, “Timing behavior of 96 radio pulsars,” in *The Astrophyisical Journal*, vol. 422, pp. 671–680, February 1994.

[28] “EPN.” www.mpifr-bonn.mpg.de/old_mpifr/div/pulsar/data/browser.html, Accessed August 2011.

[29] D. H. Staelin, “Fast Folding Algorithm for detection of periodic pulse trains,” in *Proceedings of the IEEE*, April 1969.

[30] V.I.Kondratiev *et al.*, “New limits on radio emission from X-ray dim isolated neutron stars,” in *The Astrophyisical Journal*, vol. 702, September 2009.

[31] P. McMahon *et al.*, “Real-time coherent dedispersion using FPGAs.” Poster presented at the URSI General Assembly, August 2008.

[32] M. Borga, "Canonical correlation a tutorial." www.imt.liu.se/~magnus/cca, Accessed February 2011.

[33] Xilinx, "Virtex-6 FPGA memory interface solutions," Tech. Rep. DS186, March 2011.

[34] Xilinx, "ML605 hardware user guide," Tech. Rep. UG534, July 2011.

[35] A. Cosoroaba, "Achieving high performance DDR3 data rates in Virtex-7 and Kintex-7 FPGAs," Tech. Rep. WP383, Xilinx, March 2011.

[36] The Mathworks Inc., *Simulink HDL Coder 2 User's Guide*, April 2011.

[37] Synopsys, Inc., *Synphony HLS User Guide (Actel)*, March 2010.

[38] Synopsys, Inc., *Synphony HLS User Guide*, July 2010.

[39] *Catapult C Synthesis User's and Reference Manual, UV Product, 2011a*, April 2011.