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

CAS-2012-09

M.Sc. Thesis

A Framework for Designing and Testing the Digital Signal Processing unit of a Pulsar Based Navigation System

Vigneswaran Karunanithi

Abstract

Navigation systems using pulsar signals have been of great interest ever since the discovery of pulsars in 1967 and various studies were carried out ever since to determine the feasibility of such systems. The successful working of a navigation system using pulsar would mean that spacecraft can maneuver through the deep space with very little help from earth. Another notable aspect of this navigation mechanism is that pulsar signals are available throughout the cosmos, so this system could be used for a wide range of orbital configurations such as Low Earth Orbits (LEO), Medium Earth Orbit (MEO), Geostationary Orbits (GEO) and Interplanetary orbits enabling a universal navigation system.

In this thesis, a software framework has been developed to replicate the environmental setup for a pulsar navigation system. This framework can be used to design a high level system design of the pulsar signal processing unit which will be a part of the navigation system. The framework can be used to generate pulsar templates as it can process the raw data from a receiver. The generated templates will be used in the final navigation system in the stages of correlation and detection. The software framework was developed using a very novel system design approach using SystemC-AMS, which consists of C++ classes that felicitate high level system design involving analog and mixed signals. The design approach, implementation, results and limitations of this framework is also presented. Based on the conclusions derived from the framework, a road-map is provided for future development.

TUDelft

A Framework for Designing and Testing the Digital Signal Processing unit of a Pulsar Based Navigation System

Thesis

submitted in partial fulfillment of the
requirements for the degree of

Master of Science

in

Embedded Systems

by

Vigneswaran Karunanithi
born in Bodinayakanur, 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

Cover page credits: NASA/JPL-Caltech
Cover page Description: This artist's concept depicts the pulsar planet system discovered by Aleksander Wolszczan in 1992. Wolszczan used the Arecibo radio telescope in Puerto Rico to find three planets - the first of any kind ever found outside our solar system circling a pulsar called $PSRB1257+12$.

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 **"A Framework for Designing and Testing the Digital Signal Processing unit of a Pulsar Based Navigation System "** by **Vigneswaran Karunanithi** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: 30-10-2012

Chairman:

Dr. Ir. Nick van der Meijs

Advisors:

Dr. Ir. Nick van der Meijs

Dr. Ir. Rene van Leuken

Committee Members:

Dr. Ir. Chris Verhoeven

Dr. Ir. Gerard Janssen

ir. Steven Engelen

Abstract

Navigation systems using pulsar signals have been of great interest ever since the discovery of pulsars in 1967 and various studies were carried out ever since to determine the feasibility of such systems. The successful working of a navigation system using pulsar would mean that spacecraft can maneuver through the deep space with very little help from earth. Another notable aspect of this navigation mechanism is that pulsar signals are available throughout the cosmos, so this system could be used for a wide range of orbital configurations such as Low Earth Orbits (LEO), Medium Earth Orbit (MEO), Geostationary Orbits (GEO) and Interplanetary orbits enabling a universal navigation system.
In this thesis, a software framework has been developed to replicate the environmental setup for a pulsar navigation system. This framework can be used to design a high level system design of the pulsar signal processing unit which will be a part of the navigation system. The framework can be used to generate pulsar templates as it can process the raw data from a receiver. The generated templates will be used in the final navigation system in the stages of correlation and detection. The software framework was developed using a very novel system design approach using SystemC-AMS, which consists of C++ classes that felicitate high level system design involving analog and mixed signals. The design approach, implementation, results and limitations of this framework is also presented. Based on the conclusions derived from the framework, a road-map is provided for future development.

Acknowledgments

It would not be possible to express in words how much I would like to thank my advisor Dr. Ir. Nick van der Meijs for his supervision throughout the course of this one year. The topic of radio astronomy being very new to me, I experienced many pitfalls at various stages of this thesis and if not for my advisor it would have been a tough job to get up and get going again. His help during the writing of this thesis in LaTeX is invaluable.

I would like to extend my thanks to my co-supervisor Dr. Ir. Rene van Leuken for the support and very important discussions during the modeling of the simulators.

At a very critical stage of this thesis it was because of the invaluable help from Dr. Ir. Gerard Janssen with which I could proceed further. Thankyou for the help.

At this point I would also like to thank my predecessors in this work Ir. Steven Englen (for also introducing me to this topic) and Ir. Vashishth Chaudhri for the valuable work done by them which was the starting point for this thesis.

I would like to thank the CAS group, my colleagues at the group, the Linux man of our group Antoon without whom working on the Linux platform would have been even more challenging. Though I was not part of a group project, I got the opportunity to discuss with my fellow colleagues about their work and I am very grateful to Sumeet, Anupam, Rakshith , Tasos Michos and Sachin Chadha for sharing your knowledge.

Learning a completely new tool such SystemCAMS was a challenging task and this was made easier by a number of people. Firstly, I would like to thank the online support forum SystemCAMS (which can be contacted via: ams_forum@lists.accellera.org). Specifically, I would like to extend my thanks to Sumit Adhikari from Vienna University of Technology for answering all the queries and taking additional interest in helping at important stages.

Having studied from home for over 16 years (Schooling and Bachelors degree) it was new experience to study in a completely new place and this would not have been possible but for those very important people in life called Friends. The list of friends I made in the past two years exceeds the number of friends I had made in the previous 16 years. So if I missed out mentioning all the names please accept my apologies. My dear Kuch bhi group thanks for those countless parties and the time we spent together, it was a great support.

I have had the best possible housemates in Brasserskade 23 (Vishu and Sriram) and Telderslaan 31(Naddu, Harshad, Cheshta and Malli), thankyou so much for taking great care of me during crunch times.

There has always been three very important people who have been there for me everytime and to whom I would like to dedicate this thesis to, my Mom, Dad and my

Brother. Thanks for all the support.

Thanks to the creators of the CE LaTeX thesis style for creating the template of this thesis. Without them, this would not have been possible.

Vigneswaran Karunanithi
Delft, The Netherlands
30-10-2012

Contents

List of Figures

List of Tables

Listings

List of Algorithms

I dedicate this thesis to
Amma, Appa and Vishu

Earth is a cradle of humanity,
but mankind cannot stay in the cradle for ever.

- Konstantin Tsiolkovsky

1 Introduction

1.1 Motivation

1.1.1 Navigation Systems

Since time immemorial, astronomical observations have played their parts in shaping the human evolution in their respective societies. These observations were primarily carried out as methodical observation of the night sky to predict the movement of celestial objects. The scope of these observations widened and were able to look much deeper into the universe through the invention of telescopes. From this point Astronomy could be split into two parallel branches of observation and instrumentation. Currently we have reached a stage where we can make observations from outside Earth. This trend is expected to continue as long as the human kind remains inquisitive and can exercise ways to unlock the mysteries of the universe. Considering the league of observations that can be made from outside, it is important to transport the observing instrumentation to the desired location in space and make the observation. Hence, navigation plays a key role in such missions.

Navigation in simple terms can be referred to as a process of estimating the position and velocity (state vectors) of an object as a function of time, with respect to a given coordinate system. In a space based system, this process can be carried out using two methods. The first method is known as semi-autonomous/active[1] navigation, where the probe/satellite uses external man-made beacons to identify its position. One of the most commonly used semi-autonomous navigation systems is the Global Positioning System (GPS). Navigation using GPS signals is a very effective and proven method for so many missions, but it has its short comings. As the distance from the earth increases, the GPS signals strength diminishes. Hence, beyond the LEO (Lower Earth Orbit) and partly GEO (Geosynchronous Earth Orbit), using a GPS based navigation system is simply not possible [7]. The Deep Space Network (DSN) is another well known semi-autonomous navigation technique used for deep space missions. This has served as an invisible tether between the earth and the voyager 1&2 probes.

Given that GPS based navigation is not possible for deep space missions and the DSN based navigation involves high operational costs and communication latencies, it is desired to have alternative cost effective methods which, as a result would encourage more deep space missions. One such methods is to navigate using naturally available signals and autonomously estimate the present position with respect to a coordinate system.

As the method suggests, this navigation method is classified under the Autonomous/Passive Navigation. In the context of this thesis, the work presented focuses

[1] Active and Passive refers to the amount of human intervention required to keep the system functioning.

on navigation using highly stable radio signals from pulsars. A pulsar based navigating system works similar to the LORAN-C (LOng RAnge Navigation system) [8] in terms of the position estimation algorithms used. But the source signal is the radio signals from pulsars. The method to detect a pulsar seems to be a very straightforward process considering the processing steps involved, but processing radio signals from pulsars are inherently challenging considering very low signal strengths and signal distortions induced in these signals as they travel through the stellar medium. In this thesis a framework has been developed which can be used to test the pulsar signal processing unit and to perform a high level analysis to determine the feasibility of such a system in-terms of real hardware and its usability for navigation purposes.

1.1.2 Framework development

Using pulsar signals to navigate spacecraft have eluded scientists ever since pulsars were accidentally discovered in the 1960s. Since then, the Jet Propulsion Laboratory (JPL), Naval Research Laboratory (NRL) and Defense Advanced Research Projects Agency (DARPA) have conducted various studies through 70s to early 2000s to determine the feasibility of developing a navigation system using signals from the pulsars (in both the X-Ray and radio spectrum). The results of these feasibility studies have indicated that a positional accuracy of less than 10^6 meters is achievable and it is possible to use such a system provided it could satisfy the computational needs. To test these cases it is required to develop a framework in which feasibility analysis can be performed. It is also desired that this framework closely replicates the real implementation scenario so that there is not much difference between implementable model and algorithmic model.

A pulsar navigation system, irrespective of the observing frequency consists of an analog front-end (Antennas for radio signals or X-ray detectors for capturing the photons), a detector to process the signal from the analog front-end and extract pulsar signals and a back-end system to perform navigation algorithms and provide feedback to the detector and the analog front-end. To test the detector system, it is of utmost importance to have the raw dispersed pulsar data. In most developmental processes involving interaction of multiple systems as in this case, it is required to mimic the other dependent systems so that the development process can happen in parallel. Hence, in this thesis, a major portion of the time was invested in developing a framework, to replicate the signals from pulsars and the behavior of the signal processing front-end. The signal processing front-end was designed and validated within this framework.

Another important reason for developing such a framework is that, in the future during mission analysis, this framework can be combined with the back-end models to perform real navigation. This would help in closely replicating the real scenario for very expensive missions which involve interplanetary orbits. It was decided to develop the framework using SystemC-AMS and SystemC and the framework was named as SynRaPSor (**Syn**thesized **Ra**w **P**ulsar **S**ignal generat**or** and Process**or**). In the following sections of this chapter the thesis goals/objectives and contributions of this thesis and organization of the following chapters are provided.

1.2 Thesis Goals

The primary goal of this thesis is to develop and validate an architecture for a signal processing front-end capable of processing raw pulsar data in real-time[2] and extract pulsar signals. In order to realize this goal, a working antenna with the RF front-end that can provide the raw pulsar data is required. In an early stage of this thesis, it was understood that the raw data from the actual hardware setup will not be available. So the goal was modified to developing a framework to generate this raw data mimicking the output of the antenna setup. Hence with this agenda the following thesis goals were formulated:

- **Modeling a simulation environment:** To create a software framework capable of generating pulsar signals with reconfigurable environmental parameters such as location of the pulsar, amount of dispersion, signal strength and channel noise.
- **Designing a receiver:** To design a receiver capable of de-dispersing and extracting a pulsar profile from the noisy signal transmitted by the simulation environment.
- **Validating this design:** To use this framework to process the raw data from real hardware such as the Westerbork Radio telescopes, LOFAR and possibly the actual hardware setup at TU Delft.
- **Improvement in Architecture:** To investigate the possibility of a different architecture which can provide an improvement to the present architecture.

1.3 Thesis Overview

As mentioned above the work carried out in this thesis primarily involved development of a software framework which can generate wide-band pulsar signal, simulate the channel effects caused due to the interstellar medium (ISM) and process this signal to retrieve back the original signal. To accomplish these tasks it was necessary to have a good understanding of pulsars, properties of the radio signal emitted by them and the properties of the communication channel i.e. the ISM. Following the study on pulsar the focus was shifted to instrumentation that was already available to detect pulsars.

Though there aren‘t any instrument which have been specifically made for pulsar observation with navigation as the application, instruments do exist for observing pulsars but with general astronomy as application. Hence these instruments were analyzed before moving on to developing the models for the framework. The individual models were developed and tested to check if it satisfied the properties as suggested by the literature.

[2] Real-time data processing is required as the application of this system is for navigation and hence, it is a hard requirement on the system to process the data in real-time

The individual models were integrated to form the simulators as shown in Figure 1.1 using SystemC-AMS. The integrated SynRaPSor could be considered as stages of operation as shown in Figure 1.1. The first task was to develop modules which could generate wideband pulsar signals. Then to implement channel effects which would distort the pulsar signals. Following this, the receiver to process the generated pulsar signal. Another simulator which could be considered as a subset of the SynRaPsor was developed and named as RaPSor. This simulator can be used to process the raw data obtained from real hardware. At present the simulator is configured to read the raw data from WSRT and LOFAR radio telescopes. To develop these frameworks a large amount of time was spent on studying the properties of pulsars and communication channels.

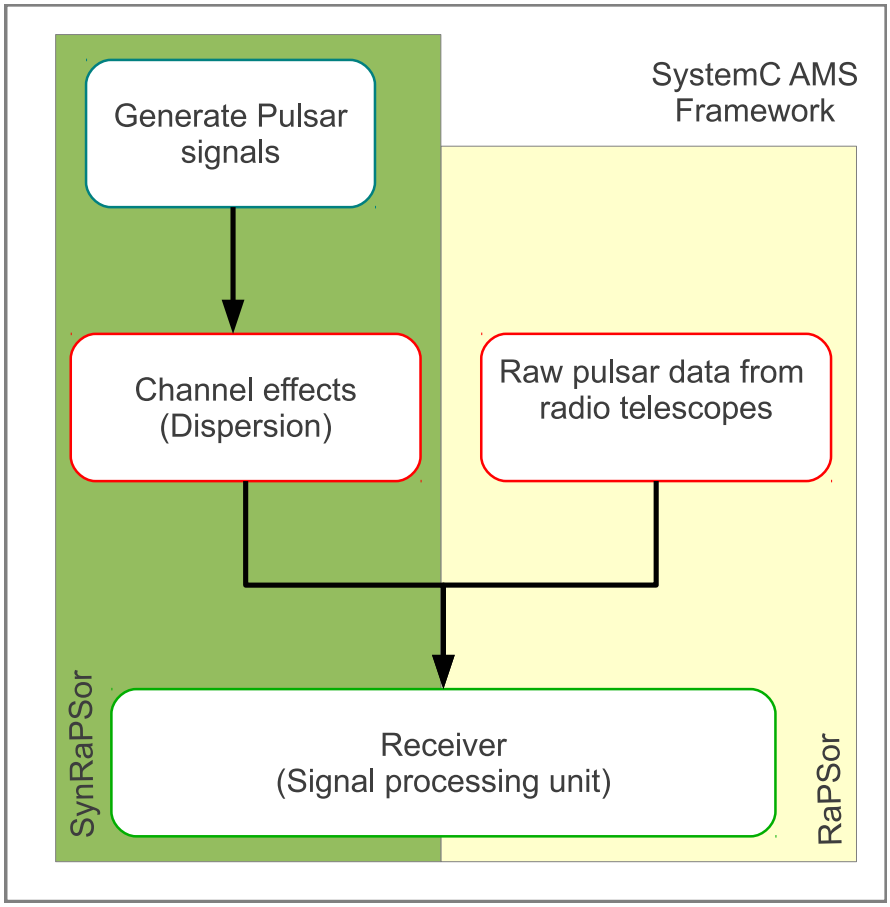


Figure 1.1: An overview of the framework developed using SystemC-AMS

Experiments were conducted using these frameworks (SynRaPSor and RaPSor) with various parameters to firstly validate the architecture of a pulsar signal processing system then processing raw pulsar signals. The findings of these experiments are elaborated in Chapter 5.

Based on the experiments carried out using the developed frameworks, conclusions

were made in order to realize a system which is feasible both in terms of processing bandwidth and computational power. These conclusions are presented in chapter 6.

1.4 Contributions

In this section, all the major contributions of this thesis are enlisted:

1. A first of its kind simulator was developed to recreate pulsar signals as received by a 10 m^2 dish using SystemC and SystemC-AMS.
2. A pulsar signal processing unit was modeled using SystemC-AMS, which can be integrated with the signal generator unit.
3. A software framework was developed to process 20 MHz bandwidth raw data (from the Westerbork Synthesis Radio Telescope) and test the performance of the signal processing receiver front end.
4. A corrected pulsar model was presented as a part of the framework which could be used to generate pulsar templates (the corrected model in reference to the work carried out in [9]).
5. A set of SystemC and SystemC-AMS libraries were created, which could be used in the future to modify the present simulator.

1.5 Outline

This thesis report is organized in two parts with the first part containing the literature survey consists of the following chapter:

Chapter 2: Pulsars, Propagation Effects and Instrumentation for Pulsar Signal Processing
Answers to some of the background questions related to pulsars, radio astronomy, navigation and an insight into the available technology for processing pulsar signals is presented in this chapter. Though there is a lot of existing literature available on this topic, it is always better to have a concise description of the topics that are most relevant to this thesis.

The second part of the thesis concentrates on the implementation aspects and is organized into the following chapters:

Chapter 3: Signal Model
In this chapter, the mathematical model of pulsars, interstellar medium and the receiver is provided. All the development work carried out in SystemC-AMS and SystemC were based on these models.

Chapter 4: Modeling
In this chapter, implementation of the models in Chapter 3 in SystemC-AMS are discussed. The mathematical models discussed cannot be implemented as it is and needs adequate interpretation to implement it in SystemC-AMS. So these interpretations and finally the implementation in SystemC-AMS are provided and following the discussion on individual components, the complete simulator consisting of these individual blocks are provided.

Chapter 5: Results, Analysis and Road-map for future development
In this chapter, all the results obtained during the course of this thesis work are presented. Following this a road-map is presented which gives a direction to the future work on this topic.

Chapter 6: Conclusion and Recommendations
The results obtained from the simulator are elaborated and recommendations to the future work that can be carried out is presented.

Appendix:
In Appendix A, a detailed description of the structure of a epn file which is the standard format to store pulsar data. In Appendix B, the SystemC based development of the receiver is discussed, which could be used for future development of synthesis-able hardware.

Pulsars, Propagation Effects and Instrumentation for Pulsar Signal Processing

2

"Following the light of the sun, we left the Old World"

- Christopher Columbus

Pulsars were first observed in the year 1967 by Jocelyn Bell. She was working with her advisor Dr.Anthony Hewish [10] as a graduate student at Cambridge University, to make radio observations of the universe. Bell and Hewish had no idea what these signals were, so they were dubbed as little green men (LGM) as a reference to extraterrestrial life and the following quote was one of the first inferences of the received signal:

"Were these pulsations man-made, but by man from another civilization?"

Soon after, Thomas Gold (a 20th century astrophysicist, from Austria) showed that a spinning neutron star could make the same pulses and this was the beginning to a new area of research in Astronomy. These pulsating neutron star were called Pulsars (Pulsating radio source). But why are pulsars such a big deal? how can it be of any help to us? and why has it been a major source of interest to radio astronomers?
Answers to these questions will be dealt with in the first few sections of this chapter, followed by an insight into the propagation effects on these pulsar signals as they travel through the interstellar medium. The next section of this chapter will be about instrumentations presently available to observe pulsars and inferences from these instrumentations to develop a system for navigation purpose. The second part of this chapter will provide an overview of the basic concepts in this thesis, a general idea of navigation is discussed. In the final part part of this chapter, a recap of the work carried out by [11] and [9] in their theses is provided with some results and recommendations from their work.

2.1 What are Pulsars ?

Stars are born in inter-stellar space through segregation of huge amounts of gas particles, which compresses due to the gravitational force. In this initial stage of formation, the cloud of dust particles called nebula, contracts due to its own gravitational force. Due to this constant compression, tremendous amounts of heat and pressure are generated. These lead to the initiation of nuclear reactions. The gravitational force of the star is towards its center and the nuclear reaction causes tremendous amount of pressure to be exerted outwards. These two forces balance each other which results in the creation of a stable star. But, in this process, the star reaches a stage where in

which the entire fuel to carry out nuclear fusion is exhausted. Hence, there will be no outward pressure to balance the inward force due to gravity. This results in the star to compress further producing even higher temperatures. Finally, the star succumbs to the heat and pressure resulting in a massive explosion (referred as Supernova). This life cycle of a star shown in Figure 2.1.

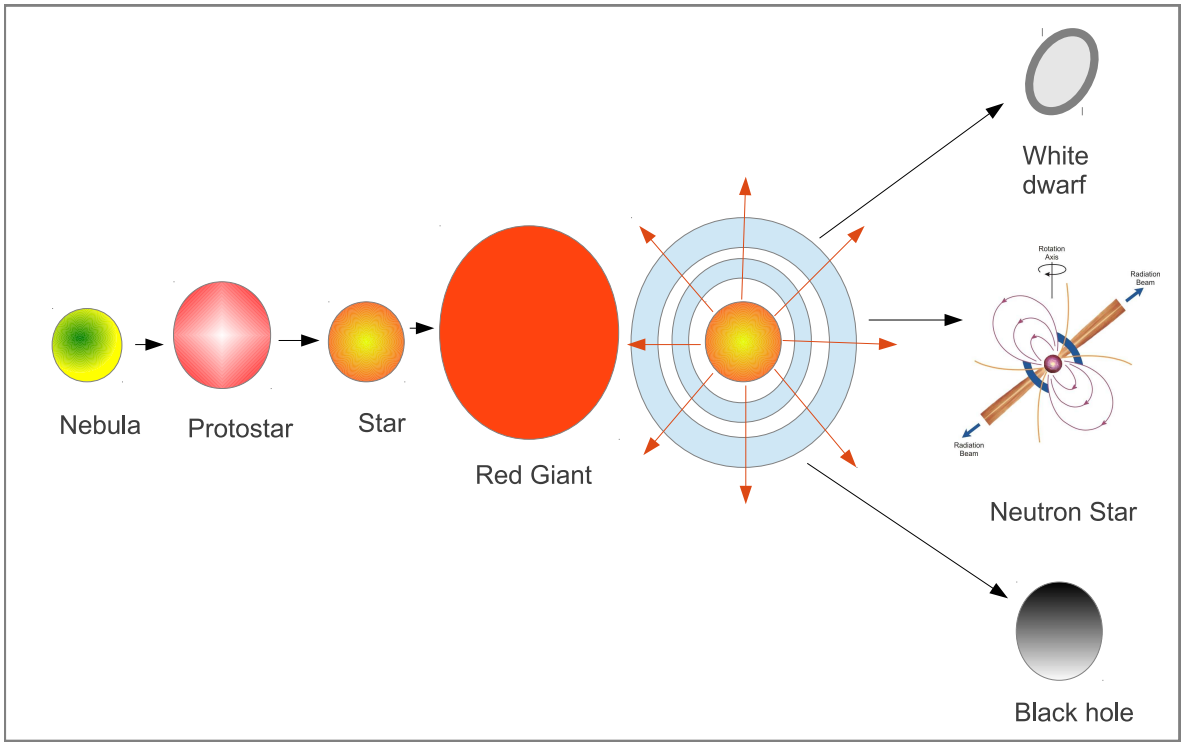


Figure 2.1: Life cycle of a star

If the mass of the star is greater than $5M_{\odot}$ ($M_{\odot}$ is the mass of our sun), the gravitational pull will be so high that the core contracts to form a black hole. The gravitational pull of a black hole is so high that even photons are absorbed. Hence there will be no radiation from this core across the entire frequency spectrum. If the mass of the core was between 1.4 to 5 $M_{\odot}$, then the star may die as a neutron star. In this case, due to the gravitational compression, the electrons are forced into the nuclei resulting in the formation of neutrons. This results in the formation of neutron stars. It has a very high mass density (mass of sun if it were shrunk in a radius of 20 km). These neutron stars conserve the magnetic flux and the angular momentum. But as the size is reduced, these effects of magnetic flux and the angular momentum are amplified many times. The resulting star emits magnetic field strengths of the order of 10^8 to 10^{15} G ($1G = 10^4T$). This induces an electric field that accelerates charged particles along the poles and as these charged particles travel through the magnetic field they produce radiation of high magnitude.

A diagrammatic representation of the magnetic field emission and the axis of rotation are shown in the Figure 2.2. It can be clearly seen from the figure that there

is a difference in the orientation of the spin axis and the radiation axis. This causes the intensity of the electromagnetic radiation to vary in a periodic fashion from a fixed line of sight. This variation is most often compared with the lighthouse effect, as the light emitted from a lighthouse is in an orthogonal axis as compared to its spin axis and when observed from a fixed location on ground, The intensity of light reaches the maximum as the radiation coincides with the observing location. The two parameters, angle between the spin axis and magnetic axis and the angular momentum is different for each pulsar and this makes the observed variation in intensities unique from one another. Figure 2.2 shows an example of pulsar profiles which correspond to the variation in intensities of nine different pulsars and it can be observed from the figure that the width of the pulsar profiles vary from one another and the other distinguishing factor for pulsars is the pulse period, which is the direct representation of the rotating period angular momentum, this period can vary from 1.4 milliseconds to 8.5 seconds [9]. Table 2.1 gives a list of the pulsar repetition frequencies for 10 different pulsars (randomly selected pulsars). The names of these pulsars are referred as PSRJ[1].

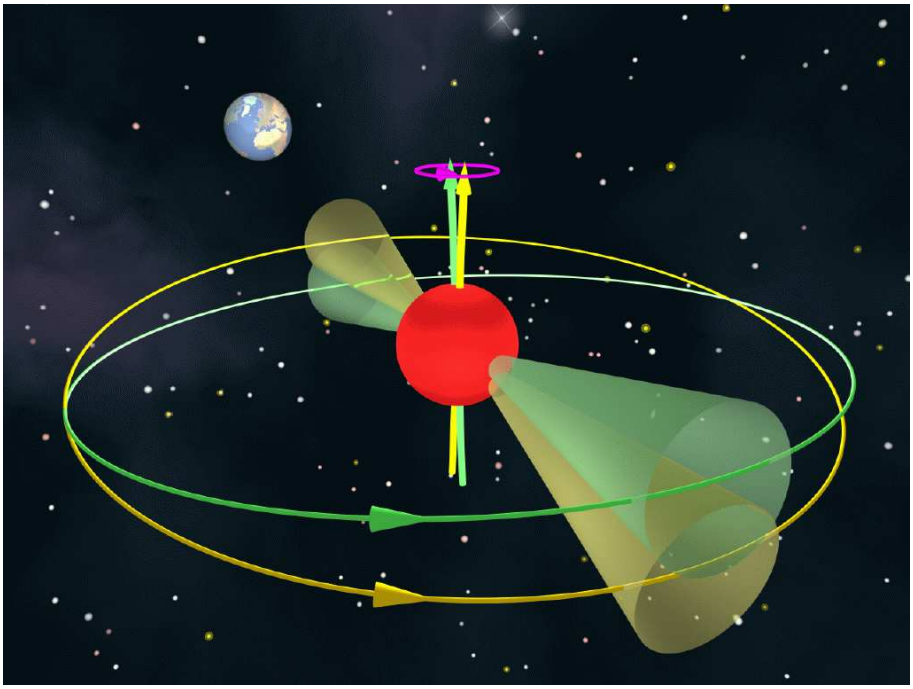


Figure 2.2: Model of a rotating pulsar [1]

2.2 Characteristics of Pulsars

It is evident from the discussions in the previous sections that the pulsars have certain properties which make them very significant objects in the space. This is why they

[1]Pulsars are either represented as PSRJ or PSRB. Pulsars which were found before 1993 were named based on their position according to B1950 coordinate system. Pulsars identified after 1993 were named based on J2000 coordinate system.

Table 2.1: A List of 10 different pulsars [6]

Serial Number	PSRJ	F0 (Hz)
1	J0006+1834	1.4414462816
2	J0007+7303	3.165827392
3	J0014+4746	0.805997239145
4	J0023+09	327.868852
5	J0024-7204C	173.708218966053
6	J0034-7204Y	1.0605004987209
7	J0040+5716	0.8942741320036
8	J0048+3412	0.821629025162
9	J1342+2822B	418.51147259915
10	J1527-5552	0.9535571091

are even sometimes referred to as physics laboratories in space. In this section, some these of these significant characteristics of pulsars will be discussed.

Periodicity: The signals from the pulsars are highly periodic.In certain cases, this periodicity is comparable to atomic clocks on earth. Though the signals from a pulsar, when observed individually vary from one another Figure 2.3, when the signal is integrated over the pulsar period, results in a stable profile. It can be seen in Figure 2.3 that the individual pulses are incomparable with one another. But when these signals are *epoch folded* (discussed later in this chapter), the result is a profile which can be easily recognized and affiliated to a pulsar. This is one of the characteristics which helps in realizing the navigation system.

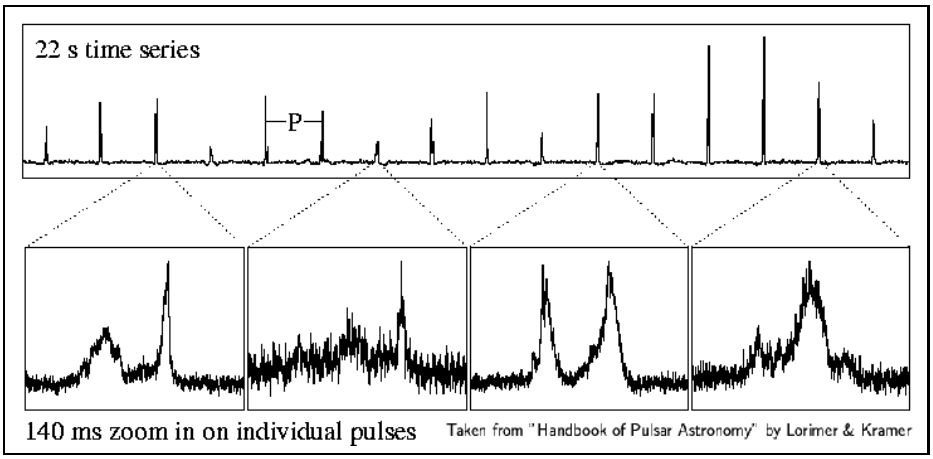


Figure 2.3: Varying intensities of the individual pulses [2]

Stability: Any moving object in space would have a slowdown and the same applies to pulsars [12] also. But, one of the advantages of the pulsar signal is that this

slow down factor (also known as the period derivative denoted by $\dot{P}$) is very low. This can be observed from the Figure 2.4.

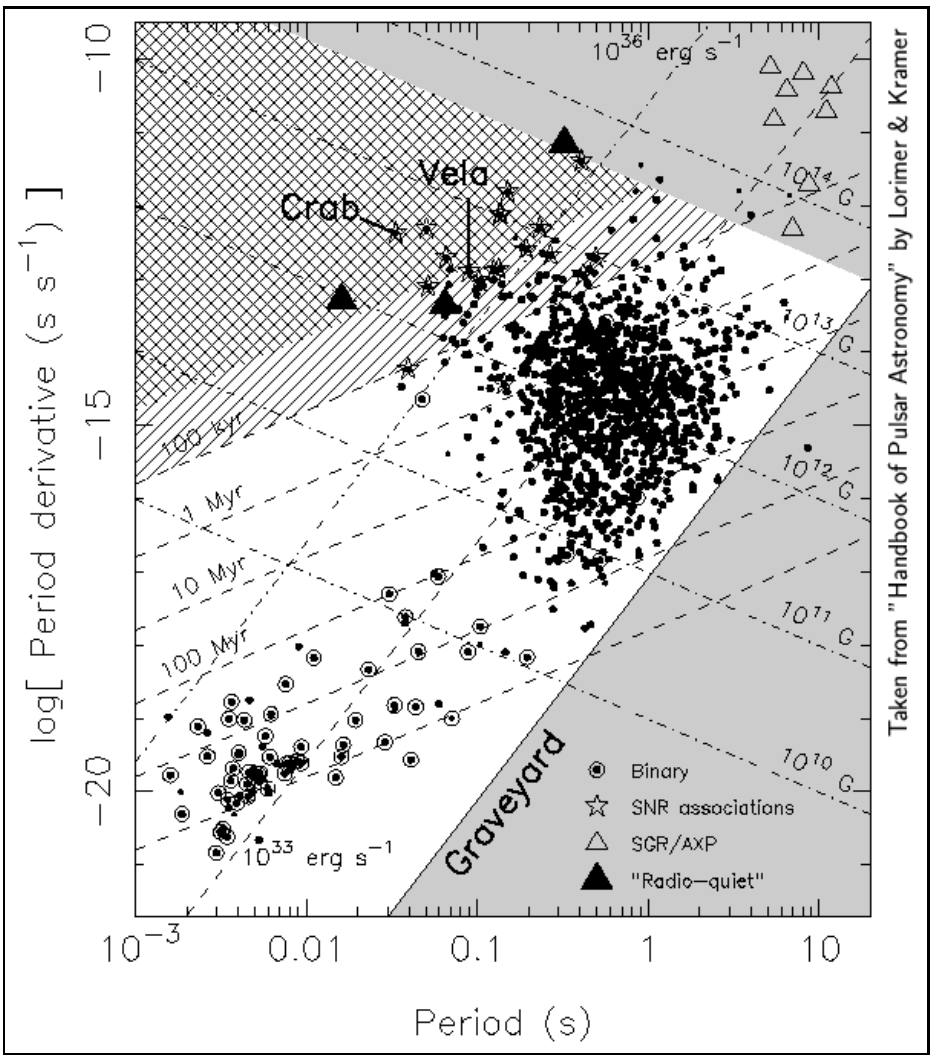


Figure 2.4: P-$\dot{P}$ gives an idea of the stability of the period of signals emitted by pulsars [2]

Frequency occupancy: The signals from pulsars occupy a very wide range of frequency band, which means that, irrespective of which frequency a receiver is tuned it will still be able to receive the pulsar signal.

Flux density: At radio frequencies, the typical pulsar flux densities range from 0.1mJy to 1Jy ($1Jy = 10^{-26}Wm^{-2}Hz^{-1}$) and from 400MHz to 1400MHz, the flux is strongest. The strongest pulsar signal recorded is at 1GHz and has a flux density of 1.7 Jy.

Distribution: The distribution of the pulsars are mainly concentrated along the galactic plane which is evident from Figure 2.5. The Figure shows the distribution of pulsars in the galactic coordinates [13].

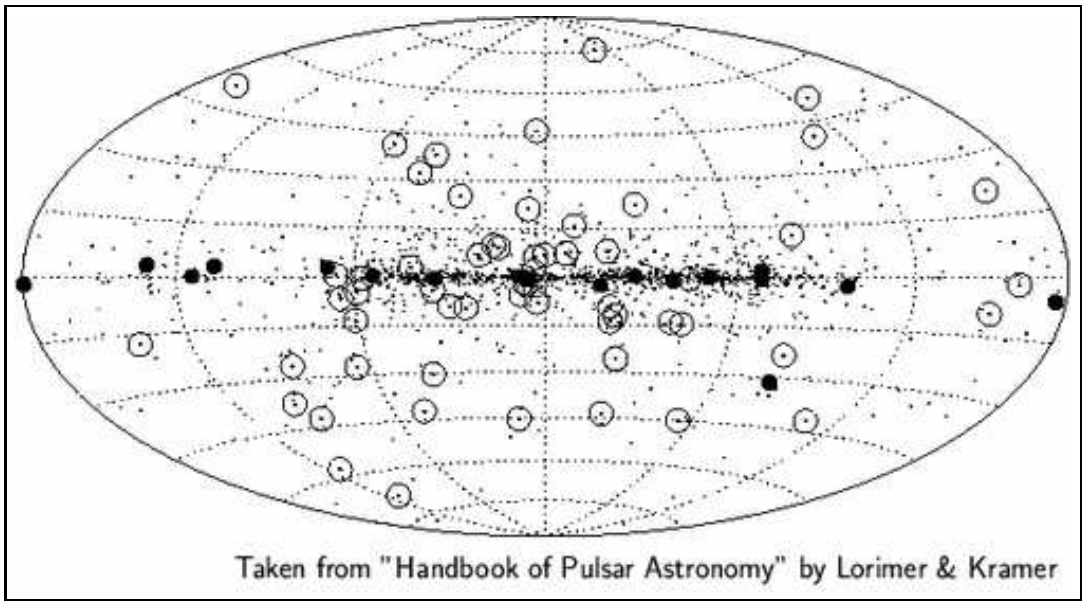


Figure 2.5: Hammer-Aitoff projection showing the distribution of pulsars in the galactic coordinate system. [2]

2.3 Propagation Effects

The pulsar signal travel a long distance through the interstellar medium before it reaches the earth. This interstellar space is a highly non-homogeneous and hence, the signal undergoes various distortions as it propagates through this medium. Three of the most distinct effects on the pulsar signal are: dispersion, scintillation and scattering. These three effects can be explained as follows:

2.3.1 Dispersion

The wide-band pulsar signal undergoes frequency dependent effects as it travels through the interstellar medium. These frequency dependent effects were first observed by Hewish in his studies carried out in [10]. The pulses observed at higher frequencies arrived earlier than the lower frequency counterparts at the telescope. This is shown in Figure 2.6. This phenomena was interpreted by Hewish as frequency dependence of the group velocity of radio waves as they propagate through the ionized part of the ISM. This concept can be well understood from the following explanation which is taken from [12]: Let the the speed of the radio waves be ν depend on the refractive index μ as $\nu = \mu c$ (c = Speed of radio waves in vacuum). The refractive index of the ISM is

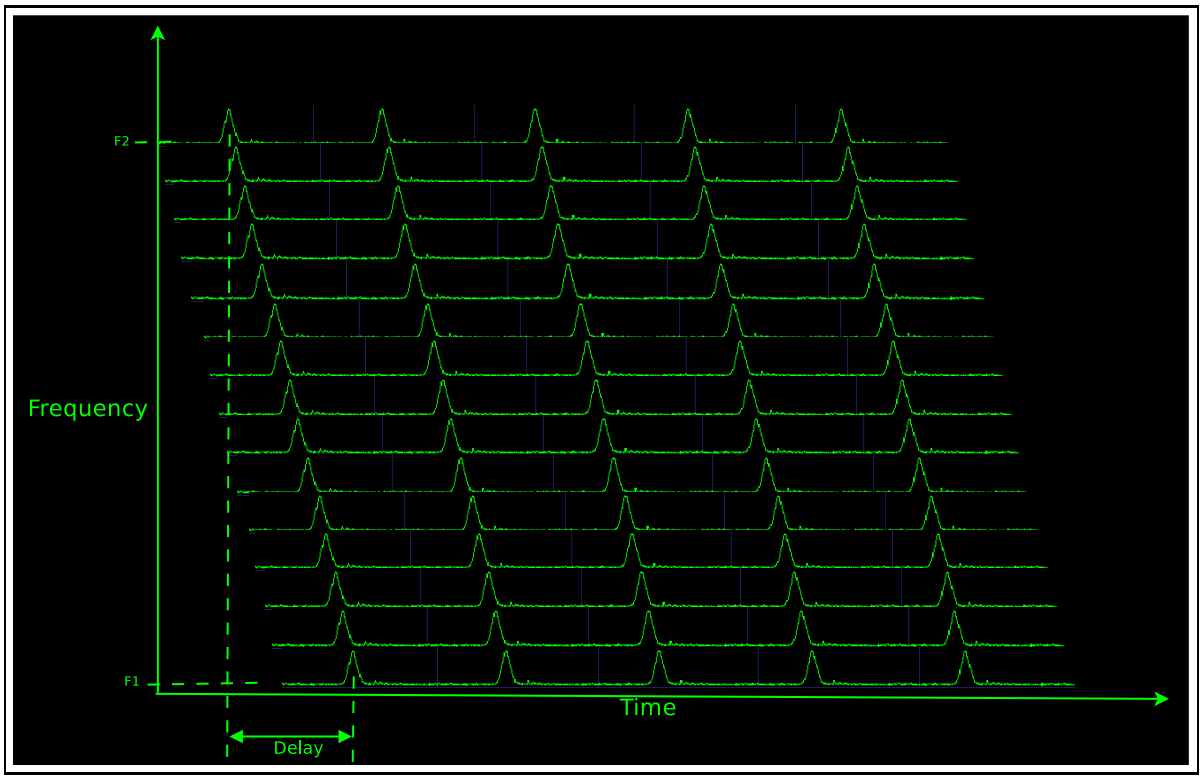


Figure 2.6: Dispersion of pulsar signal across the frequency spectrum

observed to be inversely proportional to the observing frequency f_{obs} and proportional to the plasma frequency f_p

$$\mu = \sqrt{1 - \frac{f_p^2}{f_{obs}^2}} \tag{2.1}$$

The plasma frequency f_p of the ISM is given by:

$$f_p = \sqrt{\frac{e^2 n_e}{\pi m_e}} \approx 8.5kHz\sqrt{\frac{n_e}{cm^{-3}}} \tag{2.2}$$

where e is the electron charge, m_e is the electron mass and n_e is the electron number density of the ISM, which is typically around $n_e \approx 0.03cm^{-3}$, hence corresponding to a plasma frequency of $f_p = 1.5kHz$. The observing frequency has to be greater than plasma frequency ($f_{obs} > f_p$), because if the observing frequency is lower than plasma frequency then $1 - (f_p/f_{obs})^2 < 0$ and $\nu = c\mu \notin \Re$. The total delay of the radio wave propagating along a length d is given by:

$$t = \int_0^d \frac{dl}{\nu} - \frac{d}{c} = \int_0^d \frac{dl}{c\mu} - \frac{d}{c} \tag{2.3}$$

$$t = \frac{1}{c}\int_0^d \frac{dl}{\sqrt{1-\frac{f_p}{f_{obs}^2}}} - \frac{d}{c} \tag{2.4}$$

An approximation can be made with f_p as usually the observing frequency is much higher, $f_p << f_{obs}$, hence the equation can be approximated as in $(1-x)^{-n} \approx 1+nx$ when $x << 1$

$$\frac{1}{\sqrt{1-\frac{f_p^2}{f_{obs}^2}}} \approx 1 + \frac{f_p^2}{2f_{obs}^2} = 1 + \frac{e^2 n_e}{2\pi m_e f^2} \tag{2.5}$$

Now applying 2.5 to 2.4 we get

$$t \approx \frac{1}{c}\int_0 d1 + \frac{f_p^2}{2f_{obs}^2}dl - \frac{d}{c} = \frac{1}{c}\int_0 d1 + \frac{e^2}{2\pi m_e}\frac{n_e}{f_{obs}^2}dl - \frac{d}{c} \tag{2.6}$$

$$t = \int_0^d \frac{1}{c}dl + \frac{e^2}{2\pi m_e c}\frac{\int_0^d n_e dl}{f^2} - \frac{d}{c} \tag{2.7}$$

$$t = \frac{e^2}{2\pi m_e c}\frac{\int_0^d n_e dl}{f^2} = D \cdot \frac{DM}{f^2} \tag{2.8}$$

DM is the measure of the number of the free electrons integrated along the total distance d between the pulsar and the observing point.

$$DM = \int_0^d n_e dl cm^{-3} Pc \tag{2.9}$$

D in equation 2.8 is the dispersion constant given by $D \approx e^2/2\pi m_e c = 4.1488.10^6 Hz^2 pc^{-1} cm^3 s$ The above equations can now be used to estimate the delay between two waves of different frequencies f_1 and f_2 (in MHz) as a function of DM:

$$\triangle t \approx 4.1488.10^3 \left(\frac{1}{f_1^2} - \frac{1}{f_2^2}\right) \cdot DM \tag{2.10}$$

In the context of this thesis by plugging in values in equation 2.10, with observing frequency $f_{obs} = 1200 MHz$ [9] and a bandwidth of 10 MHz, the delay between the pulse at the highest frequency(1205MHz) to the pulse at the lowest frequency(1195 MHz) is given by:

$$\triangle t = 4.1488.10^3 \left(\frac{1}{1195^2} - \frac{1}{1205^2}\right) \cdot DM = 0.00004802 * DM sec. \tag{2.11}$$

Therefore for a 10 MHz bandwidth this delay will vary between 0.04802 msec to 480.2 msec.

2.3.2 Scintillation

The interstellar medium is a highly irregular medium, in other words it is a inhomogeneous and a turbulent space with varying electron densities. These irregularities induce phase modulating effects on the propagating electromagnetic signals from pulsars [12]. As a result, the observed intensities fluctuate on a variety of bandwidths and timescales. This effect can be seen in Figure 2.7 where the flux densities of PSR B1133+ 16 vary at each of the observed frequencies. This effect is very similar to the 'twinkling' of stars when observed in a clear night sky. This effect is caused primarily by Earth's atmosphere on the visible spectrum of the signals from the stars. These effects were modeled in [14] (by Scheuer), in which the ISM was modeled as a thin screen of irregularities. Through this model Scheuer demonstrated that the intensity fluctuations should be correlated over a characteristic scintillation bandwidth $\delta f \propto f_{obs}^4$.

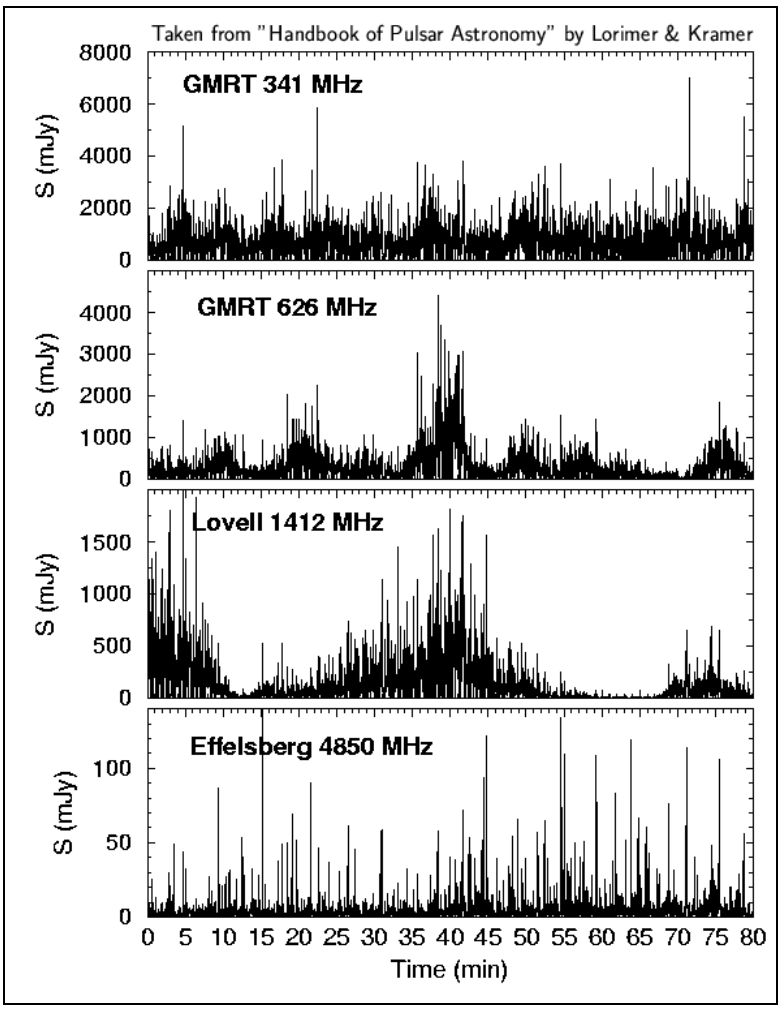


Figure 2.7: Effect of scintillation showing different flux densities of PSR B1133+ 16 when simultaneously observed at four different frequencies (Results from GMRT, Lovell and Effelsberg telescopes) [2]

2.3.3 Scattering

Due to irregularities in the stellar space the electromagnetic waves undergo scattering and travel different path-lengths. This results in broadening of the received signal which is otherwise intrinsically a sharp pulse (2.8). A very simple model can be used to predict this behavior by convolving the true pulse shape with a one sided exponential ($\frac{1}{e}$ time constant τ_s) [12]. It has been observed that the scattering time τ_s is correlated strongly with the dispersion measures DM [15]. Hence the signals from distant pulsars (large DMs) exhibit larger scattering delays, effectively stretching the pulse shapes. This effect on the pulse shape results in a reduction in S/N ratio. But it is also observed that scattering effects are reduced as the observing frequency is higher (Figure 2.8). Thus in the channel modeling of the ISM explained in (chapter 4) of this thesis, the effects due to scattering are not considered. But in the future, for more accurate models, these effects can be added to the existing framework.

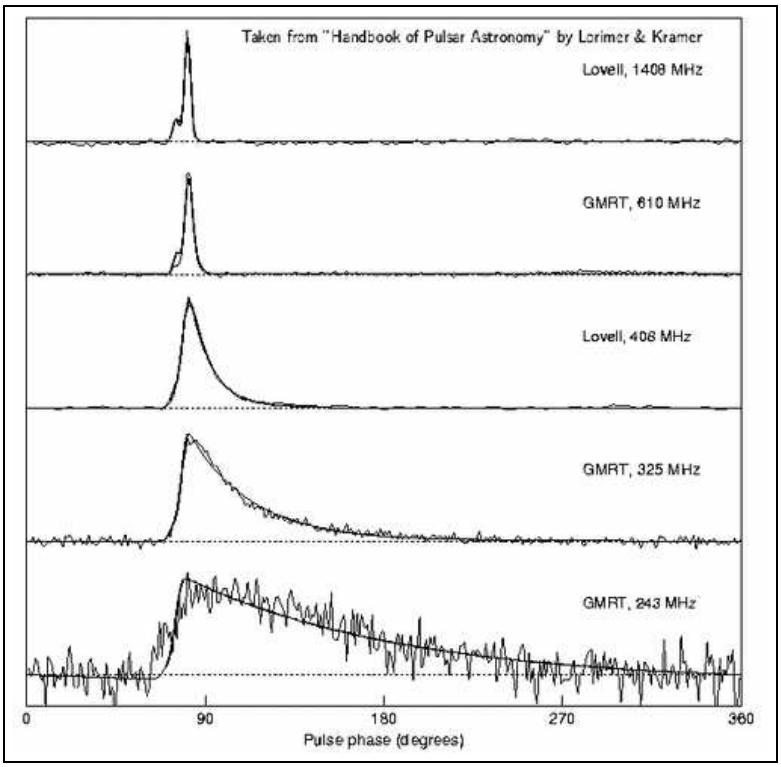


Figure 2.8: Effect of scattering on the true pulse shape, observed simultaneously at different frequencies [2]; It can be observed that the effect of scattering is lower at higher observing frequencies.

2.4 Observing Pulsars

In a pulsar based navigation system, the detection of pulsar signal is the first stage of operation. Figure 2.9 gives a general overview of the detection system and its essential

elements required to detect pulsar signals using a single radio dish. The signals received by the antenna are feed into a polarizer to provide the orthogonal components of the received signal. This weak signal is amplified using an LNA whose frequency response is centered around the f_{RF} or f_{obs} (observing frequency). The amplified signal is bandpass filtered to remove the harmonics and interferences at the observing frequencies. The filtered signal is now down converted to an intermediate frequency (IF) f_{IF} using a mixer which operates using a local oscillator at f_{LO}. The output of the mixer is a signal with frequency components $f_{RF} + f_{LO}$ and $f_{RF} - f_{LO}$ from which the difference component $f_{RF} - f_{LO}$ is considered for further processing [12].

The *IF* signal $f_{IF} = f_{RF} - f_{LO}$ can be processed straight away using analog filter

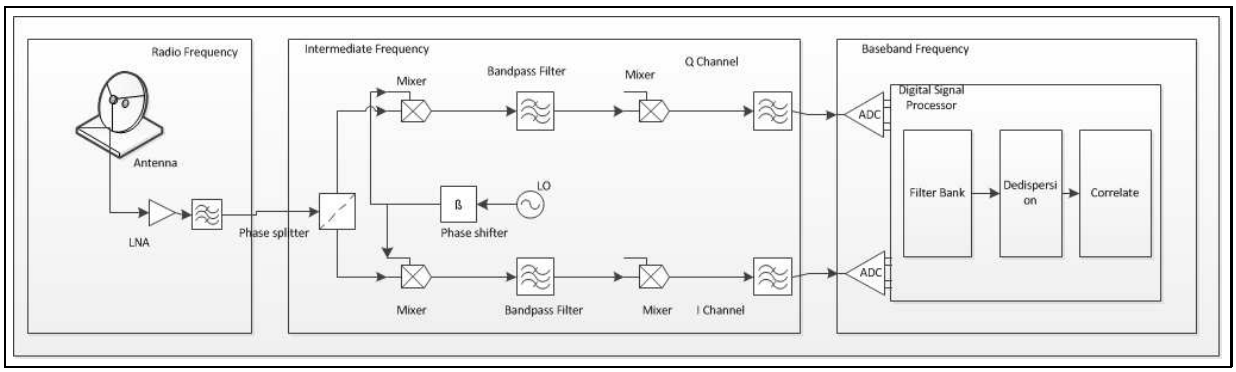


Figure 2.9: A top level system architecture of a pulsar observing system

banks and de-dispersion filters or can be further down-converted to a baseband signal for digital signal processing. The down-converted baseband signal is subjected to various signal processing blocks such as filter banks, de-dispersion filters, folding and correlators to remove the channel effects and extract the pulsar signal. All the baseband signal processing blocks which are used to process raw pulsar signals are discussed in the following sections.

2.4.1 De-dispersion

The pulsar signals undergo various propagation effects as they propagate through the stellar medium (discussed in Section 2.3) and one of the most significant effects is the frequency dependent delay induced by the interstellar medium which results in dispersion of the pulse shape. These artifacts in the received signal can be removed by the process of de-dispersion. There are two known methods to de-disperse the received signal, the first method de-disperses the signal in the time domain and is called as ***incoherent de-dispersion***, the second method employs frequency domain operations and is called ***coherent de-dispersion***. The effect of this process is shown in Figure 2.10 where the image on the left is a spectrogram view of the dispersed signal and the image on the right is the output after de-dispersing the signal completely. The above mentioned techniques of de-dispersion are discussed in the following sections.

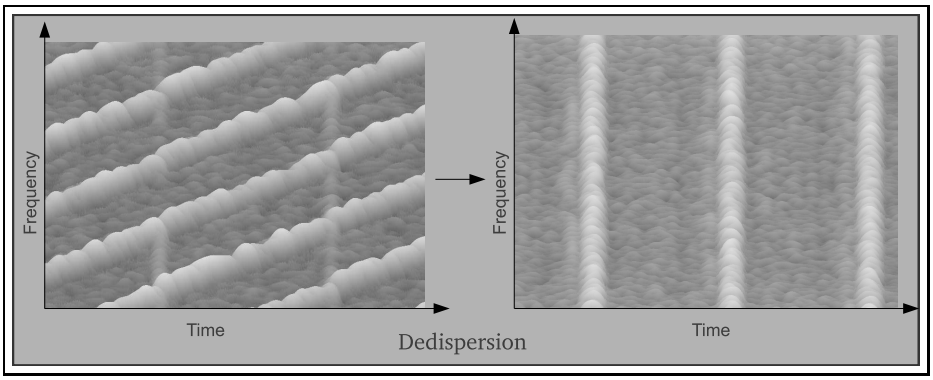


Figure 2.10: Reversing the effects of ISM through de-dispersion (Image taken from the simulation out as a part of this thesis and are discussed in the future chapters).

2.4.1.1 In-coherent de-dispersion:

In incoherent de-dispersion, the received signal of finite bandwidth Δf is split into a number of frequency channels and each channel is applied with appropriate delays to compensate for the delay caused by the ISM. The delay applied to each of the channels can be obtained from the equation 2.12.

$$\triangle t \approx 4.1488.10^3 \left(\frac{1}{f_{ref}^2} - \frac{1}{f_{chan}^2} \right) .DM \tag{2.12}$$

The resulting delayed frequency channels can be added to provide a de-dispersed signal. But in this process, the phase information of the received signal is lost, hence the name incoherent phase de-dispersion.

2.4.1.2 Coherent de-dispersion:

Coherent de-dispersion coined by Hankins and Rickett [16], involves complex frequency domain transformation of the incoming pulsar signal. In this method, the effect of the interstellar medium is described as a phase only filter, as delay in time domain is equivalent to phase shift in the frequency domain. Hence the properties exhibited by the ISM is expressed in the form of a transfer function H.
Mathematically, the expression for the transfer function can be realized as follows:
Let the observing frequency $= f_o$ and *Observing bandwidth* $= \Delta f$

$$V(f_o + f) = V_{int}(f_o + f)H(f_o + f), |f| < \Delta f/2 \tag{2.13}$$

$V(f)$: Fourier transform of the raw voltage $v(t)$
$V_{int}(f)$: Fourier transform of the raw voltage $v_{int}(t)$
As discussed earlier, the different frequency components will experience different delays. So the phase of the signal is expected to be a variable which is related to the frequency.

This relation is expressed as follows:

$$\Delta\phi = -k(f_o + f)d \tag{2.14}$$

where $k(f)$ is the wavenumber and d is the distance traveled by the pulsar signal.
The above relationship shows that the ISM modifies the phase of the signal and representing this is the form of a transfer function can be carried out as follows: The transfer function $H(f_o + f)$ is written as function which modifies the phase which corresponds to a frequency dependent exponential term.

$$H(f_o + f) = e^{-ik(f_o+f)d} \tag{2.15}$$

The derivation of the expression for this transfer function is available in [12] and the final expression obtained is shown below:

$$H_+(f + f_0) = e^{\left[\frac{i2\pi Dr^2}{f_0^2(f_0+f)}\right]}, f << f_0 \tag{2.16}$$

In the Equation 2.16, dispersion D is related to the dispersion measure DM by the following equation:

$$DM(pc.cm^{-3}) = 2.41.10^{-4}D(sMHz^2) \tag{2.17}$$

The inverse of the transfer function H^{-1} is then multiplied with the input stream of raw voltages in the frequency domain.

$$V_{int}(f) = V(f)H^{-1}(f_o + f) \tag{2.18}$$

In practical implementations the $H^{-1}(f_o + f)$ has to be combined with a taper function T, which is chosen such that it emphasizes anti-aliasing in low pass filtering. Following are the steps required to perform coherent de-dispersion [12]:

1. The received signal is down converted as a baseband and the received I and Q channel signals are digitized by sampling at a frequency $2BW$.

2. The digitized signal is processed in segments, so the first n samples of the received signal are used to create as a data set and passed processed further.

3. The data segment containing the n samples are fast Fourier transformed and multiplied with the inverse transfer function of the interstellar medium.

4. The segment of data which was multiplied with the complex coefficients of the ISM are inverse Fourier transformed to get the coherently de-dispersed data.

The computational requirements of a coherent de-dispersing unit [12] are very high and this can be explained as follows:
For an observing bandwidth Δf with a sampling frequency f_{samp}
The sampling time is given by $t_{samp} = 1/f_{samp}$
The sampled signal needs to be observed for a duration proportional to the DM before processing it in the frequency domain given by:

$$t_{obs} \propto DM \propto \Delta f \tag{2.19}$$

The number of point FFT required to process this observed signal is proportional to the observing time given by:

$$n_{FFT} = t_{obs}/t_{samp} \propto DM * \Delta f^2 \tag{2.20}$$

The time required to perform a single FFT operation is proportional to

$$t_{FFT} \propto n_{FFT} log_2(n_{FFT}) \tag{2.21}$$

Therefore, the ratio of computational time to observing time which represents the computational complexity involved is given by:

$$t_{FFT}/t_{obs} \propto \Delta f * log_2(DM\Delta f^2) \tag{2.22}$$

2.4.2 Folding

The signal strength of the signals received from pulsars are very low and lie in the range from 0.1mJy to 1Jy and at this signal strength, it is not possible to identify the periodic variations in the intensities from the pulsars alone. Hence to detect that the received signal is from a particular pulsar will need certain integration time. Another reason why a signal has to be integrated over a timespan is that, the individual pulsar profile shapes vary, but upon integration over a timespan the pulse profile tends result in a steady shape. Figure 2.11 gives the values of the received SNR as a function of frequency for 15 best Pulsars which were measured with an antenna of effective area of $10m^2$.

This process of integrating the received signal to improve the SNR is called as folding, the received raw signal is folded over a period of time to cancel out the noise. After a certain integration period, the pulsar's profile is obtained. The signal-to-noise ratio of a pulsar with pulse period T is given by the ratio of the average spectral power $\omega_{P_T} WHz^{-1}$ and noise power $\omega_N WHz^{-1}$.

2.5 Pulsar Signal Processing Instrumentation

A pulsar signal processing unit meant for navigation using radio signals from pulsars doesn't exist till date. In order to develop such a system, the only other close choice is to investigate instruments used in observatories to observe pulsars. In this section, we will have look at some existing earth based observatories, which employ signal processing to receive radio signals from pulsars. The purpose of these systems are for observing pulsars with greater accuracy hence the size and capability of these systems will be no match to our system. But this study was done, mainly to understand the processing steps involved and draw conclusions from existing architectures. A primary observation that was made, w.r.t signal processing for detecting pulsars is that, maximum resources are spent in de-dispersing the received signal across the given bandwidth (both in-terms of hardware and software). Hence in the following systems, the de-dispersion stage of the signal processing will be mainly discussed.

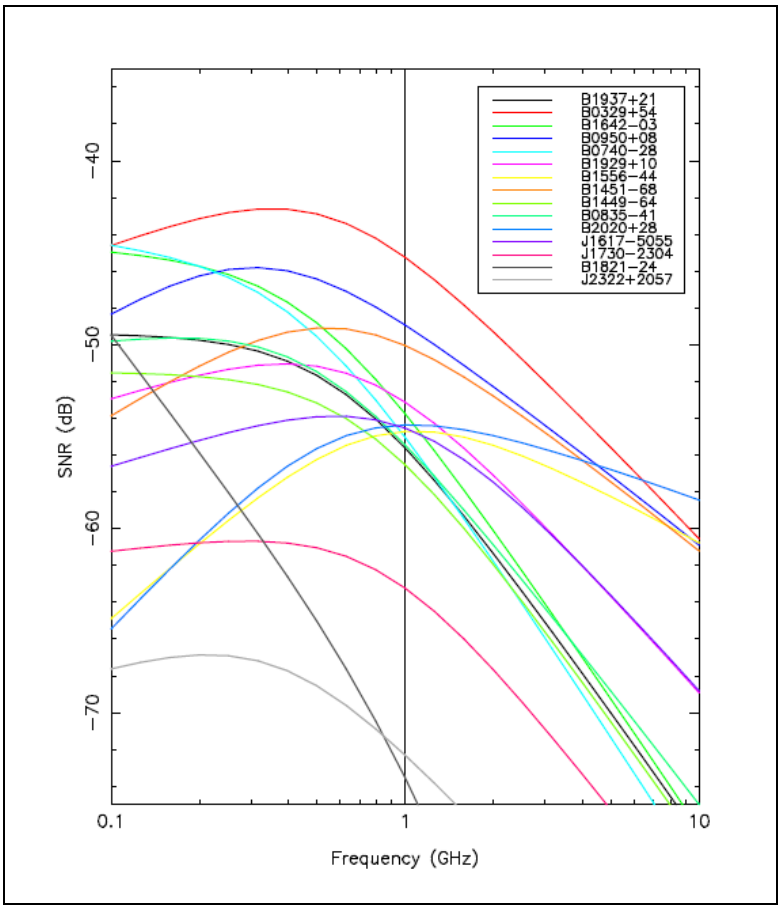


Figure 2.11: SNR(in dB) as a function of frequency of 15 best Q pulsars (Effective area = $10m^2$, beam efficiency = 0.9) [3]

2.5.1 Princeton MARK 4/5

The Princeton MARK VI, is an instrument designed to coherently de-disperse signals for pulsar observations. This system is at present installed at the Jodrell Bank (Arecibo radio telescope), the following Figure 2.13 shows the hardware architecture of MARK VI. It implements coherent de-dispersion with the help of DSPSR libraries through software. The system can accept intermediate frequency (IF) signals of Bandwidth 5 or 10 MHz centered at 30 MHz [4]. These voltages are mixed to down convert to the baseband producing two orthogonal polarizations, a real and imaginary signal with passband 0 to B/2. These four signal are low pass filtered and then 4-bit digitized at rate B. A data rate of 10 MB/s is maintained for this system. This data is processed using the software libraries off-line by a SPARC-20 machine.
The system produces packed data at the rate of 10MB/s which translates to 35 GB/hr. This rate is the input rate to the system. Hence the processing i.e. unpacking, de-dispersion and folding should be done at a rate comparable to the input rate, for the system to work real-time. For this purpose, 1.25 Gflops parallel processor optimized for FFT is used.

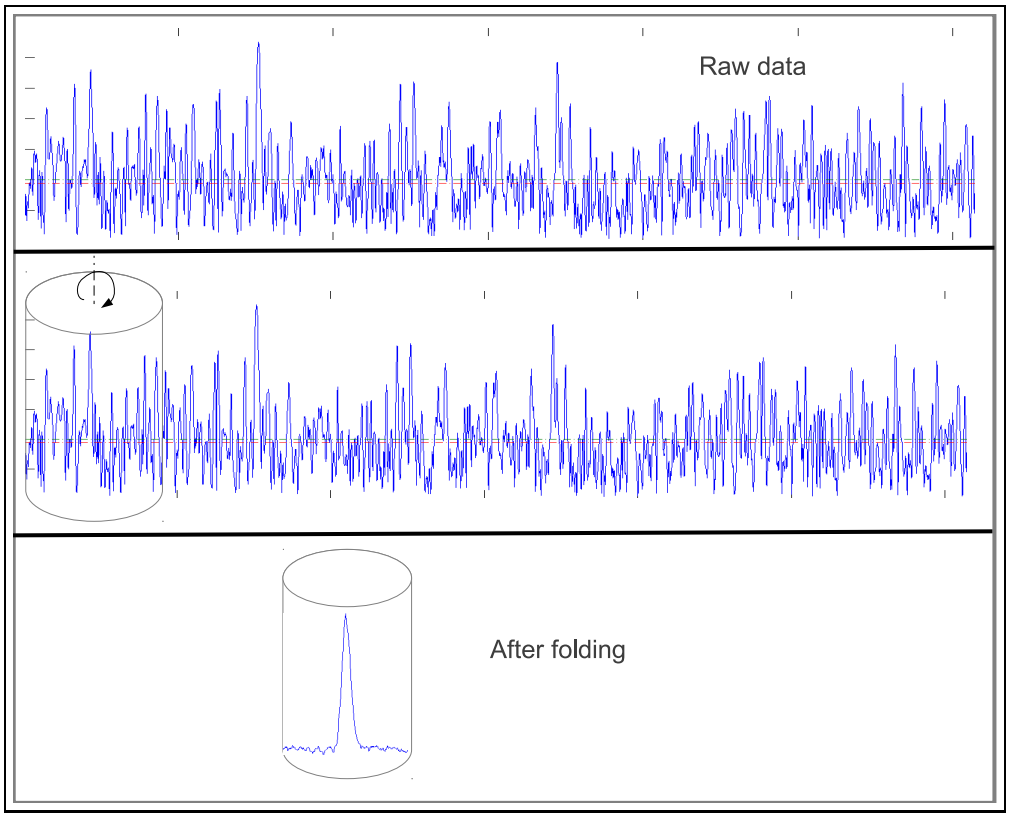


Figure 2.12: Folding of a weak pulsar signal results in a strong signal

2.5.2 Pulsar Machine II (PuMa)

Westerbork Synthesis Radio Telescope (WSRT) located in the north of Netherlands consists of an array of 14 dishes of 25 meters each. PuMa-II is a pulsar signal processing system, to process the data collected by the WSRT. It is one of the most advanced pulsar signal processing systems available, which is capable of processing a large bandwidths. Following are the system specifications of PuMa-II [17][18]:

- The received bandwidth of 160 MHz can be precessed as 8x20 MHz sub-bands. It employs phase coherent de-dispersed.
- The system achieves a time resolution of 25 ns and capable of reaching up-to 3ns.
- Data is handled as 8-bit dual polarized data with a data rate of 40 MSamples/sec.
- Distributed data processing capability: 32 processing nodes are used to perform the signal processing.

Following diagram shows the hardware architecture of PUMa-II:

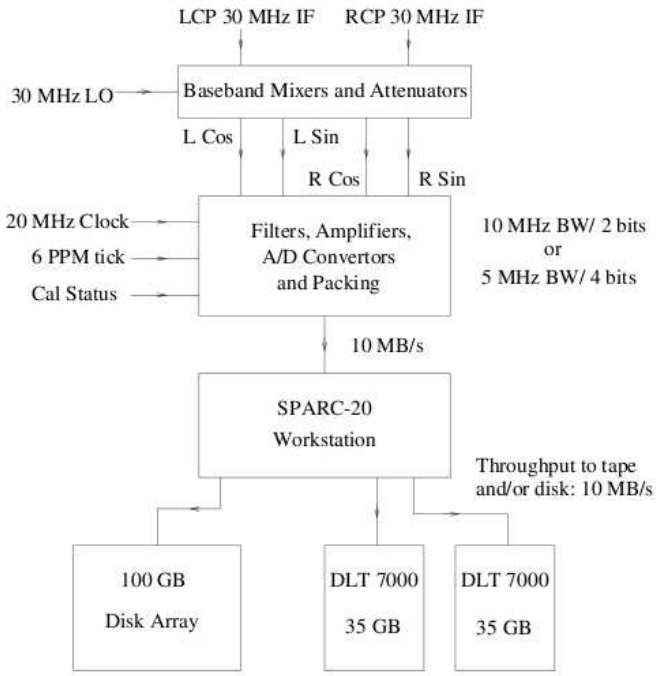
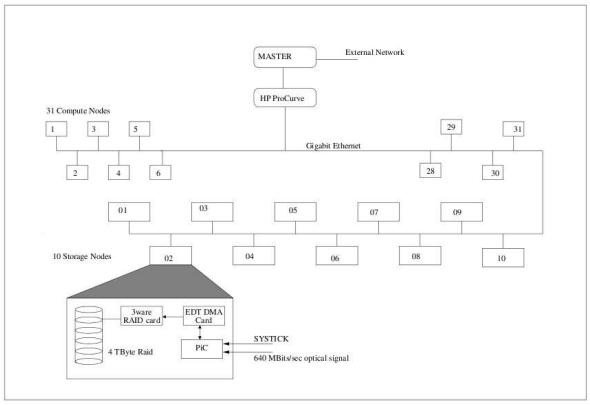


Figure 2.13: MARK VI Hardware Architecture [4]

(a) Westerbork Radio Telescopes



(b) PuMa II Hardware Architecture [18]

Figure 2.14: Westerbork Radio Telescope and PuMa-II

2.5.3 Greenbank Ultimate Pulsar Processing Instrument GUPPI

GUPPI is different from other instruments because, it makes use of graphics processing units (**GPUs**) to perform state of the Art Digital Signal Processing to coherently de-disperse in real time. The GUPPI is the first member of a family of pulsar machines built in Green Bank with the reconfigurable computing elements and supporting design tools designed by the Center for Astronomy Signal Processing and Electronics

Research (CASPER) and collaborators. The link [19] gives details of all the open source resources available from CASPER GUPPI made use of the following hardware boards from CASPER.

- iADC : An Analog-to-Digital converter of the input signal [20].
- IBOB : Internet Break out Board, which packetizes the data [20].
- BEE2 : Berkeley Emulation Engine 2, high performance signal processing [20].

Some of the expected performance parameters of GUPPI are as follows:

- Capable of processing a baseband signal in the bandwidth range of 5 MHz to 1000 MHz.
- Make use of an 8-bit digital sampler.
- A Polyphase filter bank capable of splitting the input frequency bandwidth into 256 - 4094 frequency bands.
- Have a 50 microsecond of integration time.

2.6 Summary of existing systems

From the study on the different radio pulsar signal processing units following are the conclusions drawn from the existing earth based systems.

- Almost all of the above mentioned systems make use of very large computational power with the objective of processing large bandwidths. The received signal sensitivity is directly proportional to the square root of the bandwidth and the Antenna gain G:

$$S \propto G(G = \frac{A_{eff}}{2.K_B}) \tag{2.23}$$

  A_{eff} is the effective aperture of the antenna.

$$S \propto \sqrt{\Delta f} \tag{2.24}$$

  But for space-borne system, there are two limitations. First, the is no room for large receivers which allow higher sensitivity. Secondly, there is not enough power to process large bandwidths. Hence, the sensitivity is reduced by many folds as compared to these systems.

- The coherent de-dispersion technique is computationally very intensive and the resolution derived from this method is not required for most of the applications. So a method where, combining filter-banks and coherent de-dispersion can be used to remove dispersion in an efficient way.
- The size of the FFT engine: If n be the number of samples corresponding to the time for the signal to seep across the bandwidth, then each sample will depend on $n/2$ samples before it and after it. Hence the length of the FFT should be $2n$ and a power of 2.

2.7 Digital Signal Processing

In this section, some of the concepts of digital signal processing which will be used in the context of this thesis are discussed. It is imperative to highlight these concepts before moving further to the system modeling and system design. One of the subdivisions of digital signal processing techniques is the multi-rate digital signal processing. In multi-rate digital signal processing, the sampling rate of a signal is varied (increased or decreased) during signal processing operations. In the context of pulsar signal processing, very high bandwidths are required to be processed. Multi-rate processing techniques prove to be very efficient in such applications which involve high processing bandwidths. Two of the primary operations or building blocks used in multi-rate processing are decimation and interpolation.

- Decimation: The process of reducing the sampling frequency f_s of a signal $x[n]$ by a factor in M: Down-sampling.
- Interpolation: The process of increasing the sampling frequency f_s of a signal $x[n]$ by a factor in L: Up-sampling.

In the above mentioned operations, the sampling frequencies of the original signals are altered. This inherently calls for the use of filtering techniques which have to be used in conjunction with the decimation or interpolation operations to avoid aliasing. In the following sections, the signal processing blocks mentioned above are elaborated. These multi-rate techniques are used to device a poly-phase decomposition filter, which is used to channelize the incoming wide-band data-stream into smaller frequency bands. This filter is explained in the following section.

2.7.1 Poly-phase Decomposition

One of the methods to de-disperse the received dataset is to extract sub-bands of the received bandwidth and process these individual bands. In signal processing this is termed as sub-band processing and this approach helps in reducing the processing bandwidths, which directly relates to the size of the FFTs to be taken. The process of channelizing the received bandwidth into smaller sub-bands involve the use of multi-rate signal processing discussed above and in the case of a poly-phase filter bank a decimator. As the sampling rates are changed, it is necessary to have low-pass filtering to avoid aliasing.

2.8 Work done in the past

The conceptual design of a navigation system using radio signals was proposed in [9] and [21]. In the proposed design, the system was divided into two parts which were called as the "front-end" and the "back-end". The front-end, in the context of this thesis refers to the detection stage which process the baseband signal to extract the pulsar signal. The back-end system, is used to control the front-end and implement navigation algorithms. Following the conceptual design, specifications formulation, architecture and a system design approach using HDL coder was carried out in

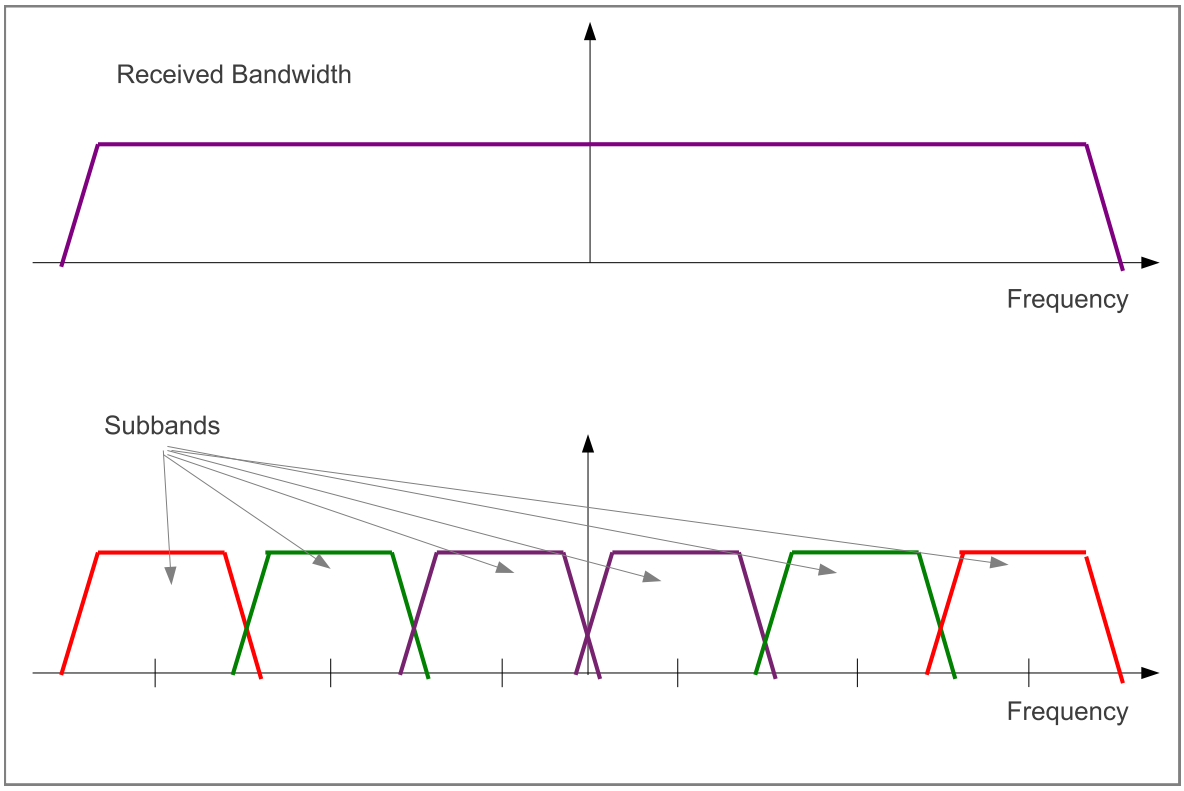


Figure 2.15: Polyphase decomposition of the input bandwidth

[11]. Some of the important findings of this thesis work are revisited in the following list:

- The frequency of observation for the navigation system was decided to be a higher frequency band (conventionally 400MHz and the 1400MHz bands are used) as signals from pulsars closer to the galactic plane suffer higher background noise at lower frequencies.
- A 12 bit ADC was concluded to be suitable for this application.
- Folding operation provided an improvement in the SNR of 20 dB which is not adequate for this application.
- Coherent de-dispersion technique was chosen to de-disperse the received signals as it provided an improvement factor upto 50 to 75 dB.
- The processing bandwidth was set to 30 MHz.
- In the system design approach followed, it was observed that using a HDL coder to develop this application had several limitations (listed in [11])

With findings from the above mentioned concepts the requirements were formulated and further development work was carried with the objectives mentioned in Chapter 1.

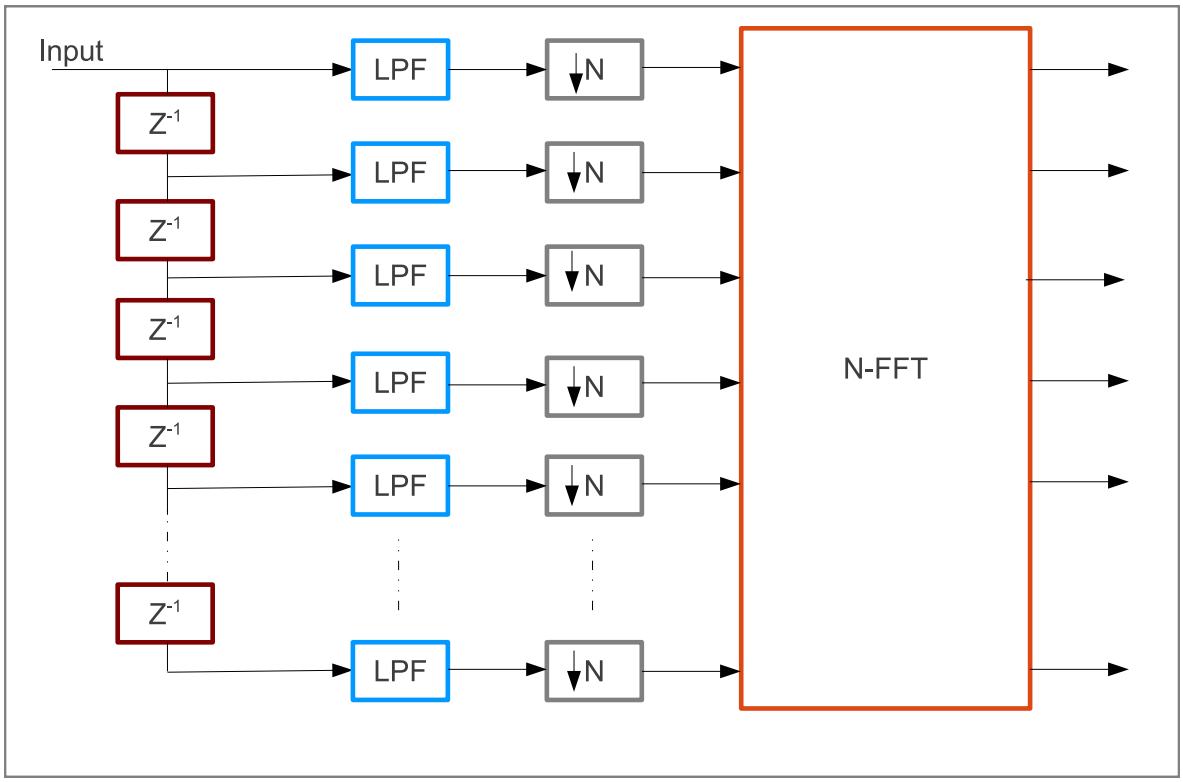


Figure 2.16: Polyphase filter structure

Figure 2.17 gives the baseline architecture which is considered for the front-end portion of the navigation system.

2.9 Requirements of a pulsar signal processing Front-end

In this section, requirements for a Signal processing front-end for a radio pulsar navigation system are listed:

- R1: The final front-end system should be capable of processing at-least a bandwidth of 30 MHz.
- R2: The system should process the 30 MHz bandwidth by dividing the it into smaller bandwidths.
- R3: The system should accommodate to the memory bandwidth needs.
- R4: The Received signal should be de-Dispersed across the 30 MHz Bandwidth.
- R5: The system should be able to fold the signal at flexible time periods, as required by the back-end[11].
- R6: The system should be able to perform cross-correlation of the folded signal and be able to detect and lock to a pulsar signal.

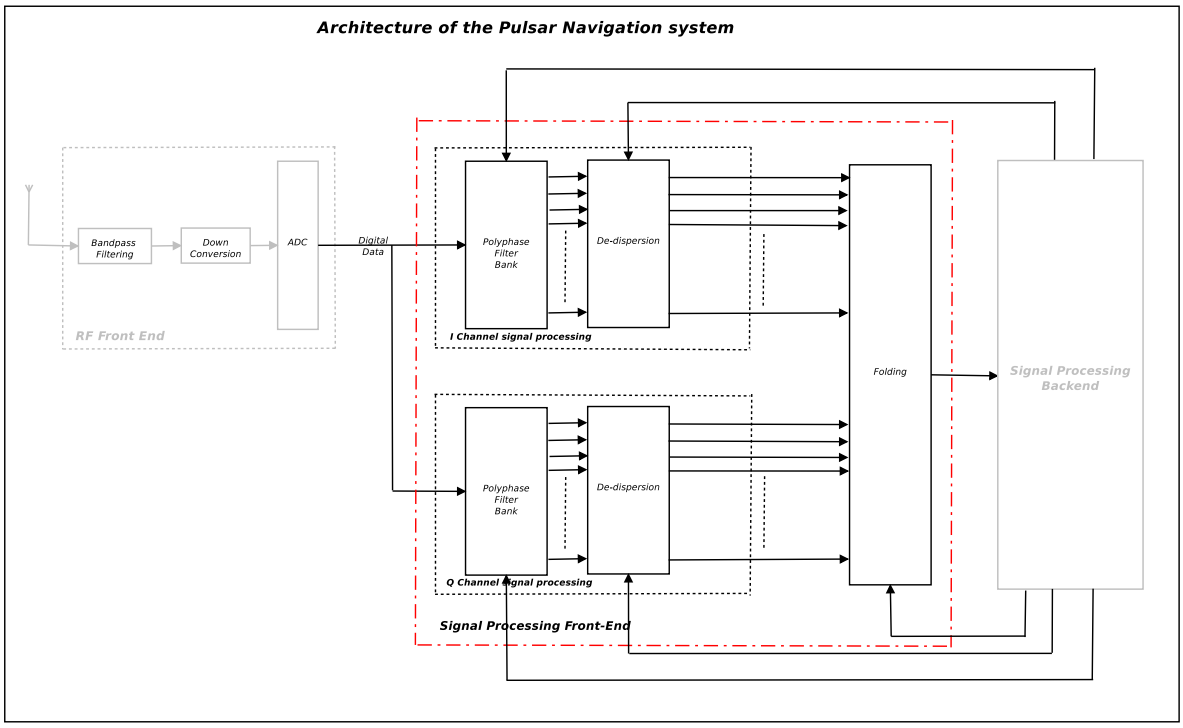


Figure 2.17: The Baseline architecture of the front-end module of the navigation system as derived from the previous work

- R7: The final system should have resolution ¡ 3m for navigation purpose.
- R8: Speed Requirements: To be determined
- R9: Power Requirements: To be determined

2.10 Conclusion

Pulsars are celestial objects which have peculiar properties making them a very interesting object to observe for astronomers. Till now signals from pulsars have primarily been used to study the evolution of our universe and to perform these studies, some of the most powerful antenna networks and computing systems have been used. Even using such high-end systems, it takes several hundreds of seconds of integration time to resolve the received signals to a particular pulsar. But considering an ideal scenario, the properties of pulsars make it a suitable candidate for navigation in deep space as they can be uniquely identified. From the sections discussed above, it is also evident that the bottleneck in designing a system for navigation using pulsars will be the processing requirements of such a system which is expected to be on-board a deep space probe adhering to the power and area requirements of the probe. As a result some very important trade-offs have to be made between increasing processing capability to get higher resolution and decreasing the required integration time to identify a pulsar.

Another aspect that was concluded from the literature studies is that for testing a processing unit, there was no simulation environment to test the processing unit during the design stage and development using tools such as Simulink didn't prove to be very efficient. Hence, it was intended to first develop an environment where the processing blocks can be tested and also use an alternate system design approach as compared the one followed in [11].

3 Mathematical Model

In this chapter, the mathematical models of pulsars, channel and a signal processing receiver are introduced. It is necessary to understand the modeling decisions mathematically before moving on to the implementation details of the individual models. The mathematical models were developed based on the work in [3] as a starting point. These models were then developed as a high level system design using SystemC and SystemC AMS, which is discussed in the next chapter.

3.1 Modeling Pulsar Signal as a broadband signal

Modeling the pulsars is equivalent to modeling a transmitter unit in a communication system. Here the transmitter can be considered as a system which emits a signal which can be recognized across a wide range of frequencies and the intensity of this signal pulsates periodically with time (Pulsar Repetition Period: PRP). Let $v_{int}(t)$ represent the initial signal from a pulsar as shown in Figure 3.1. From the properties mentioned

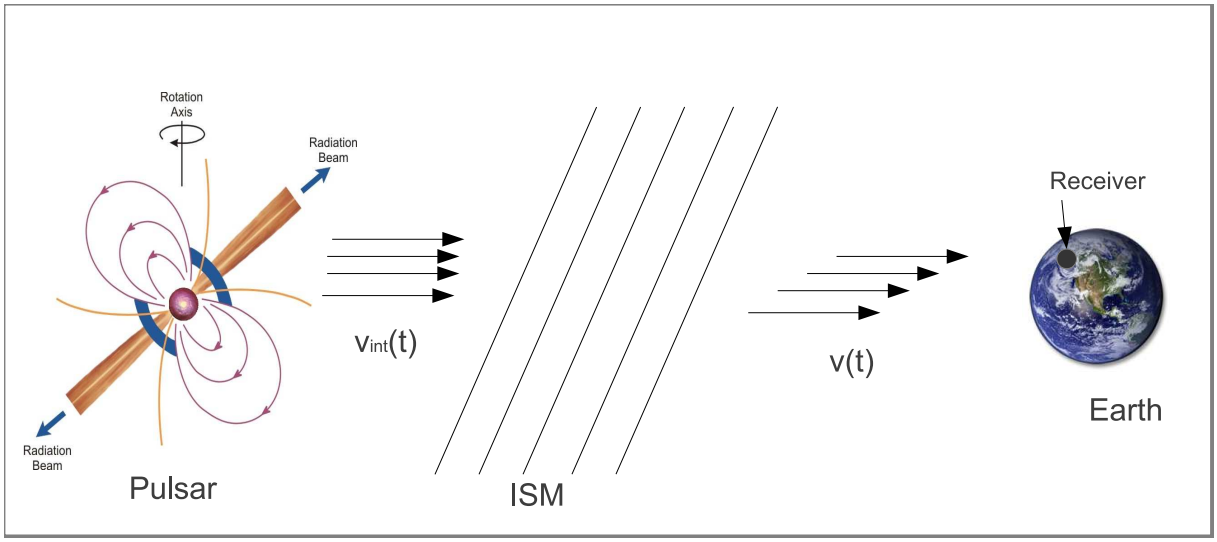


Figure 3.1: Components of the signal model

above, $v_{int}(t)$ can be modeled as a Cyclo-stationary signal. Cyclo-stationary refers to the periodic variations of the signal statistics. An example of a stationary process is an autocorrelation function $r_{xx}(t,\tau) = Ex^*(t)x(t+\tau) = r_{xx}(\tau)$. This process is a function of delay τ, but in a cyclo-stationary process $r_{xx}(t,\tau)$ depends on τ and is periodic in time. This satisfies the property of periodicity of the transmitted signal. The signal also needs to be a wide band signal, hence the stationary process is modeled as random noise

whose variance is a function of a periodic signal. The model of the transmitted pulsar signal considered doesn't take into account the slowdown or orbital perturbations on the signal. These features can be added in the future for much more accurate results. Let $P_k(t)$ represent the signal in each emission period (when observed from a fixed point), then from the assumption made above that this is a cyclo-stationary signal, a single emission can be represented as $P_k(t) = x(t + kT)\Pi(t/T)$ ($\Pi(t) = 1$ in $|t| \leq \frac{1}{2}$ and $\Pi(t) = 0$ in $|t| > \frac{1}{2}$) is a broadband pulse. Hence the broadband pulse $v_{int}(t)$ is expressed as:

$$v_{int}(t) = \sum_{k=-\infty}^{+\infty} P_k(t - kT) \tag{3.1}$$

The mean power profile of $P_k(t)$ is defined as:

$$\begin{aligned} \sigma_p^2(t) &= E|v_{int}(t + kT)|^2.\Pi(t/T) \\ &= E_k|P_k(t)|^2 \end{aligned} \tag{3.2}$$

3.1.1 Multiple Transmitters

The cyclo-stationary process stated in the previous section is a representation of the signal from one pulsar but in a real life scenario, pulsars are spread across the sky and signals received could have more than one pulsar's signal present. Hence to replicate this scenario a multi-pulsar(multiple transmitter) model is considered. This is an extension to the previously described model. A diagrammatic representation of this scenario is shown in Figure 3.2, where there are multiple transmitting sources called Pulsar1, Pulsar2 and Pulsar3.

Now considering a general case where the number of pulsars equal N then each of the

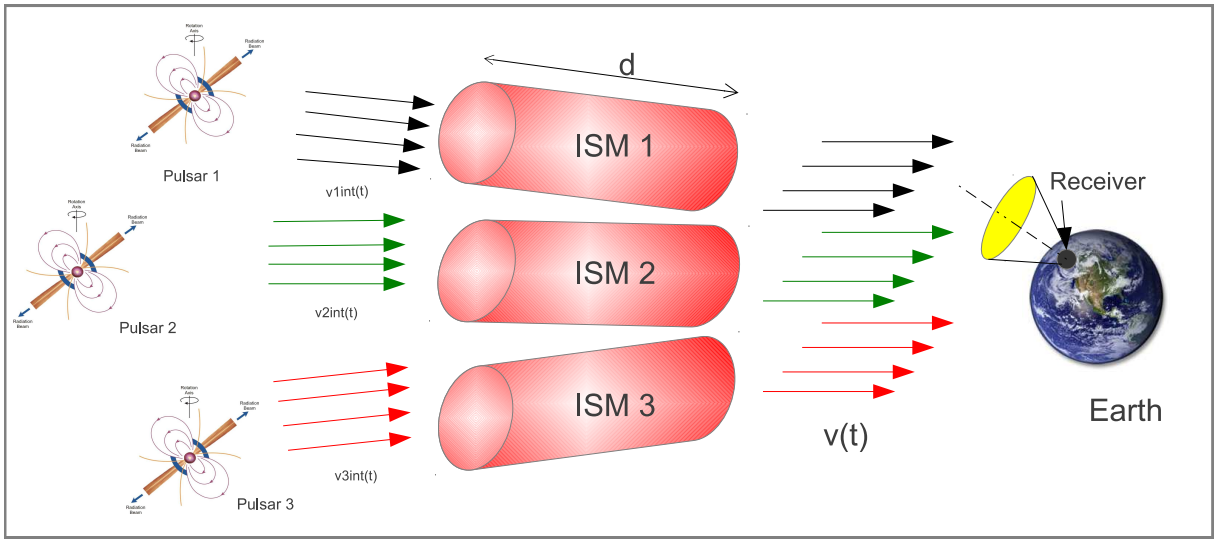


Figure 3.2: Multiple pulsars transmitting at the same time

pulsars have their own pulsar repetition period T_1 T_2 T_3 and T_N.
The initial signal $v_{int}(t)$ from each of these pulsars can be considered as $v_{int1}(t)$ $v_{int2}(t)$ $v_{int3}(t)$ to $v_{intN}(t)$ is given by:

$$v_{int_1}(t) = \sum_{k=-\infty}^{+\infty} P_{k_1}(t - kT_1) \tag{3.3}$$

$$v_{int_2}(t) = \sum_{k=-\infty}^{+\infty} P_{k_2}(t - kT_2) \tag{3.4}$$

$$v_{int_N}(t) = \sum_{k=-\infty}^{+\infty} P_{k_N}(t - kT_N) \tag{3.5}$$

and,

$$\sigma^2_{p_1}(t) = E|v_{int_1}(t + kT_1)|^2.\Pi(t/T_1) = E_{k_1}|P_{k_1}(t)|^2 \tag{3.6}$$

$$\sigma^2_{p_2}(t) = E|v_{int_2}(t + kT_2)|^2.\Pi(t/T_2) = E_{k_2}|P_{k_2}(t)|^2 \tag{3.7}$$

$$\sigma^2_{p_N}(t) = E|v_{int_N}(t + kT_N)|^2.\Pi(t/T_N) = E_{k_N}|P_{k_N}(t)|^2 \tag{3.8}$$

Each of these signals can be modeled as independent white process with spectral shaping at unique pulse repetition periods. At this stage these individual pulsar signals can be considered as independent signals which travel through the ISM independently. In the next section, the effects of the interstellar medium on the individual pulsar signals are discussed.

3.2 Channel Model of the Interstellar Medium

The individual pulsar signals in reality travel through the interstellar medium and undergo dispersion as they travel through the medium. The amount of dispersion undergone by a pulsar signal is directly proportional to distance traveled by the pulsar signal. Hence to model this phenomena, the pulsar signals can be considered as signals traveling through individual columns of electrons which undergo different dispersion levels (related to the distance traveled by each pulsar signal).

Due to this dispersion in each of the columns, the signal traveling through the column undergoes a delay at each frequency component defined as,

$$T(f) = T_\infty - D(d)f^{-2} \tag{3.9}$$

where $D(d)$ is the dispersion constant which is related to the distance traveled d. In the present model, this dispersion is realized by performing a frequency domain multiplication of the input data-stream in each column with the corresponding dispersion filter $H(f + f_o)$ derived in chapter2. Hence, this process can be modeled as frequency

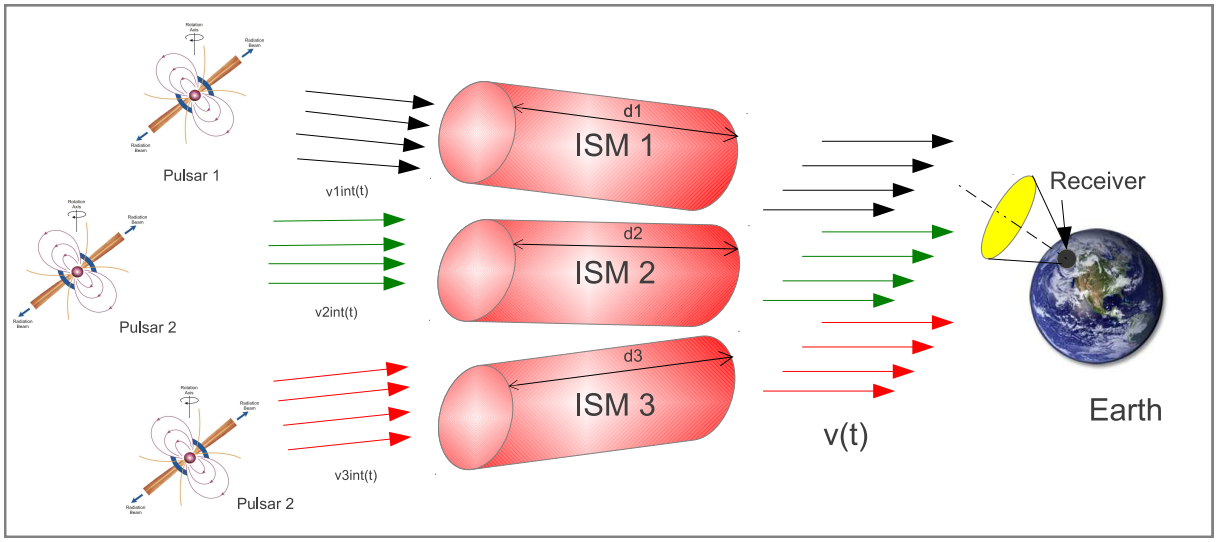


Figure 3.3: Individual columns of electrons which a length equal to the distance between the pulsar and the receiver

domain multiplication of the $v_{int}(t)$ signal with the coefficients of the filter $H(f + f_o)$ which represents the transfer function of the ISM as shown in Figure 3.4.

$$v_1(t) = IFFT[V_{int_1}(F)H_1(F + F_o)]; \tag{3.10}$$

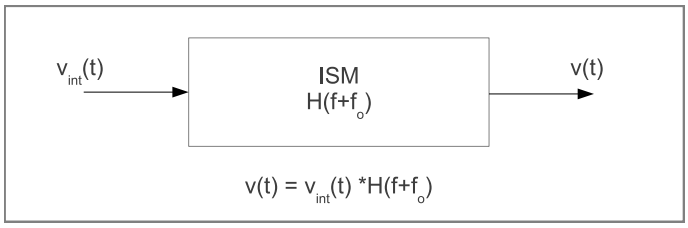


Figure 3.4: The ISM modeled as a simple convolution operation

3.3 Mathematical Model of the Receiver

The signal received by the receiver is through an antenna with a limited gain, aperture and field of view. Hence if the antenna is pointed straight at a pulsar, the signal strength is maximum. As the pointing alignment between the antenna and the pulsars changes the received signal strength also changes. In this model, the antenna is assumed to have only one lob in the forward direction with a field of view of θ^o hence signals from only those pulsars within this field of view can be received. This concept is shown in the Figure 3.5 where the receiver is located at the origin and is pointed towards pulsar P1

with and azimuth and elevation of α_1 and β_1. Now considering the model of the pulsar signal and the channel model described in the previous sections, the signal available to the receiver is the sum of the individual pulsar signals. The received signal:

$$v(t) = v_1(t)(cos(\alpha_1)cos(\beta_1)) + v_2(t)(cos(\alpha_2)cos(\beta_2)) + ... + v_N(t)(cos(\alpha_N)cos(\beta_N)) \tag{3.11}$$

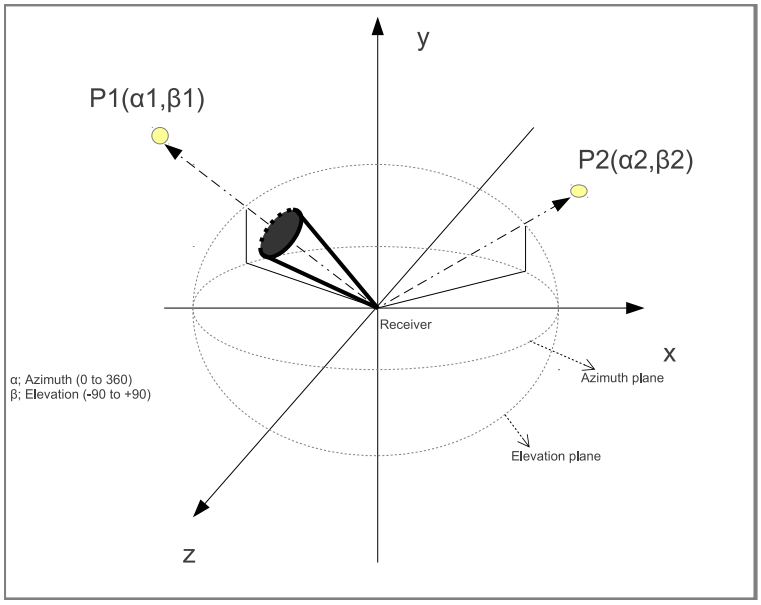


Figure 3.5: F.O.V of the receiver

The received band-limited signal $v(t)$ can be written as a combination of its amplitude, a time varying phase term and the carrier frequency centered at f_o.

$$v(t) = a(t)e^{i\phi(t)}e^{i2\pi f_0 t} \tag{3.12}$$

This received signal is mixed with the signal from a local oscillator (LO) of frequency f_{LO} to produce a signal I and a quadrature component is produced by mixing the received signal with a 90^o phase shifter version of the f_{LO} signal.
The resulting I and Q channel signals can now be represented as follows:

$$I(t) = a(t)e^{i\phi(t)}e^{i2\pi f_o t}cos(2\pi f_{LO}t) \tag{3.13}$$

$$Q(t) = a(t)e^{i\phi(t)}e^{i2\pi f_o t}sin(2\pi f_{LO}t) \tag{3.14}$$

Applying Euler's expansion, we get

$$I(t) = \frac{1}{2}a(t)e^{i\phi(t)}\{e^{i2\pi(f_o+f_{LO})t} + e^{i2\pi(f_o-f_{LO})t}\} \tag{3.15}$$

$$Q(t) = \frac{1}{2i}a(t)e^{i\phi(t)}\{e^{i2\pi(f_o+f_{LO})t} - e^{i2\pi(f_o-f_{LO})t}\} \tag{3.16}$$

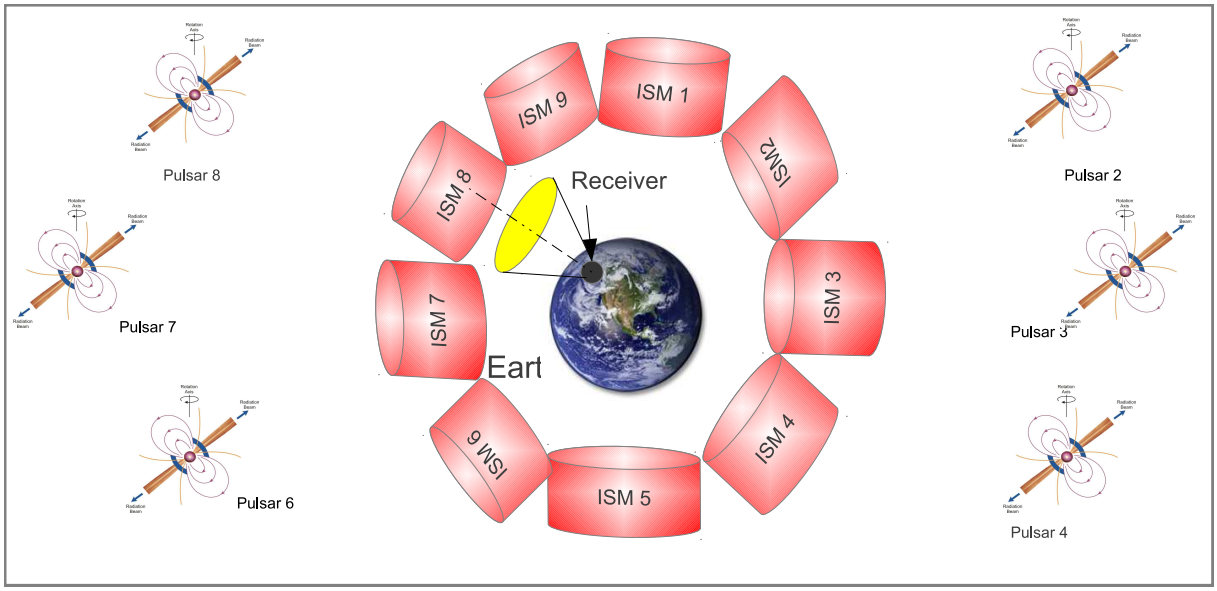


Figure 3.6: Receiver with a restricted field of view

These signals contain both $f_o + f_{LO}$ and $f_o - f_{LO}$ hence these signals are passed through a low pass filter to remove the $f_o + f_{LO}$ components hence resulting in

$$I(t) = \frac{1}{2}a(t)e^{i\phi(t)}\{e^{i2\pi(f_o - f_{LO})t}\} = \frac{1}{2}a(t)e^{i(2\pi(f_o - f_{LO})t + \phi(t))} \tag{3.17}$$

$$Q(t) = \frac{i}{2}a(t)e^{i\phi(t)}\{e^{i2\pi(f_o - f_{LO})t}\} = \frac{i}{2}a(t)e^{i(2\pi(f_o - f_{LO})t + \phi(t))} \tag{3.18}$$

The frequency of the local oscillator is selected such that the resulting signal is a baseband signal. Hence $f_{LO} = f_o$. The above equations reduce to the following

$$I(t) \equiv Re(I(t)) = \frac{1}{2}a(t)cos(\phi(t)) \tag{3.19}$$

$$Q(t) \equiv Re(Q(t)) = -\frac{1}{2}a(t)sin(\phi(t)) \tag{3.20}$$

3.3.1 Sampling

The analog baseband signal is discretized by sampling using an analog to digital converter(ADC). The sampling frequency $f_s = 1/T_s$ of the ADC has to be such that it is agreeable with the Nyquest Criterion. Hence as a rule of thumb the minimum required operation frequency of the ADC is $2BW$ where BW is the observing bandwidth. The discretized signal $I[n]$ and $Q[n]$ and be expressed as

$$I[n] = I(nT_s) \tag{3.21}$$

$$Q[n] = Q(nT_s) \tag{3.22}$$

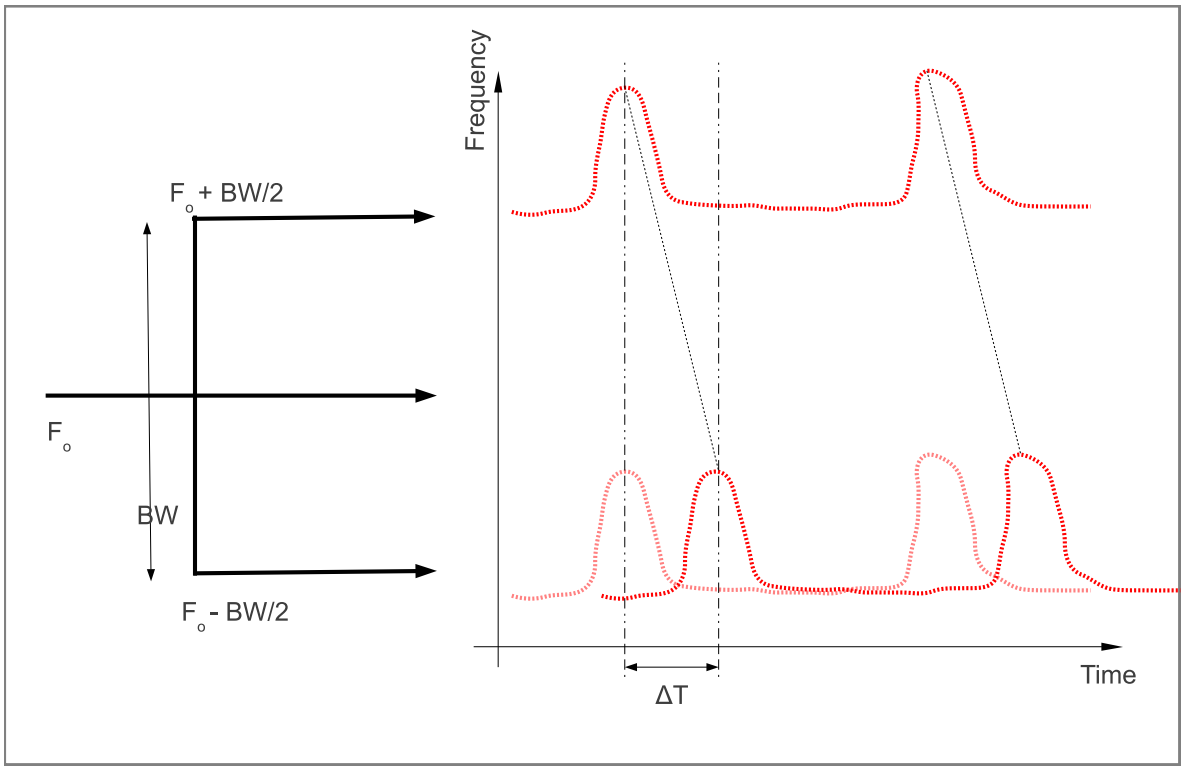


Figure 3.7: Spectrogram representation of the processing bandwidth

3.3.2 Poly-phase decomposition

The sampled signal needs to be decomposed into smaller frequency bands and this is performed using a poly-phase filter bank. An M-channel polyphase filter can be considered as follows:
Let $H(z) = \sum_{n=-\infty}^{\infty} h(n)Z^{-n}$ be the transfer function of the digital filter. Then h(n) can be decomposed into odd and even components as

$$H(z) = E_e(z^2) + E_o(z^2) \tag{3.23}$$

where,

$$E_e(z) = \sum_{n=-\infty}^{\infty} h(2n)Z^{-n} \tag{3.24}$$

$$E_o(z) = \sum_{n=-\infty}^{\infty} h(2n+1)Z^{-n} \tag{3.25}$$

The above grouping is for a two-component polyphase decomposition. This grouping can be extended to an M-component polyphase decomposition as:

$$H_o(z) = \sum_{k=-0}^{M-1} Z^{-k} E_k(z^M) \tag{3.26}$$

here the polyphase components are given by:

$$e_k(n) = h(nN + k) \tag{3.27}$$

On applying the incoming samples of I[n] and Q[n] will result in the following output:

$$\sum_{k=0}^{M-1} I_k[n + k] \tag{3.28}$$

and

$$\sum_{k=0}^{M-1} Q_k[n + k] \tag{3.29}$$

This equation can be considered as each signal $I_k[n + k]$ and $Q_k[n + k]$ represents a subband of the actual bandwidth of I[n] and Q[n].
These individual subbands are now processed using the $H^{-}1(f + f_o)$ filters with the new bandwidth reduces by a factor of M.

3.4 Conclusion

In this chapter, the mathematical models of each of the individual components of the framework were discussed. These models will be used as a guideline to implement these components using SystemC and SystemC AMS C++ libraries. The details of implementation are provided in the following chapter.

System Modeling using SystemC-AMS

4

The signal processing front-end consists of three main processing stages as shown in Figure 4.1. In the first stage, the received signal is corrected for the effects caused due to the frequency dependent refractive index of the interstellar medium on the wide-band pulsar signals. This is named as the De-dispersion stage. Following it, the received signal is integrated over the pulsar period as the received signal strength are very low. This stage is named as Folding. The integrated signal is correlated with an existing database to validate the presence of a pulsar signal in the received data (for an elaborate description of these concepts refer to Chapter 2, Pulsar Instrumentation). For designing and testing this system (shown in Figure 4.1), it is essential to provide it with a dispersed signal mimicking the signals from actual pulsars. It is desired to vary the parameters of the generated pulsar signal with varying pulse repetition period, signal bandwidth, dispersion measure and channel noise. This will help in quantifying the performance of the receiver. Hence, two simulators were developed in this thesis, which are named as **Syn***thesized* **Ra***w* **P***ulsar* **S***ignal generat***or** *and Process***or (SynRaPsor)** *Processor and* **Ra***w* **P***ulsar* **S***ignal Process***or(RaPSor)**. *Features of these simulators would be discussed in the following sections of this chapter.*

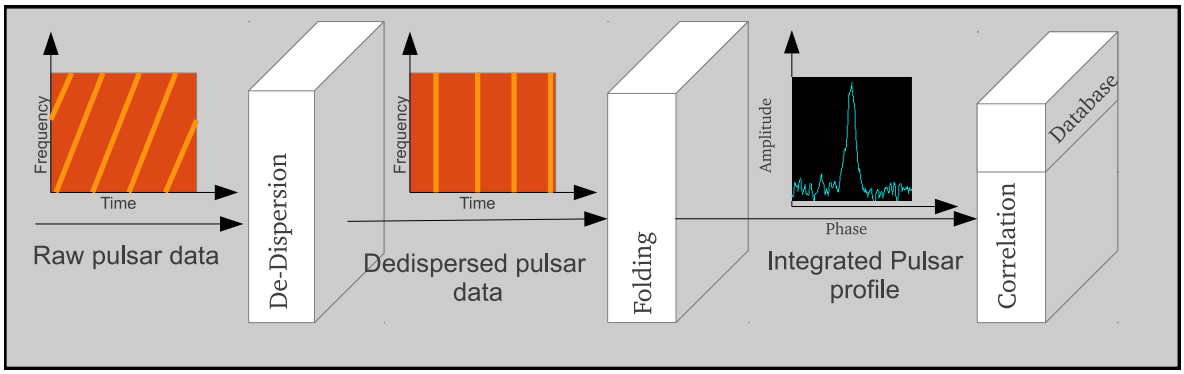


Figure 4.1: Baseline functional blocks of the signal processing front-end

4.1 Introduction to System Modeling using SystemC and SystemC-AMS

Most of the modern electronic systems usually consist of both Analog/RF and Digital components interacting with each other. Modeling and validating the high level system specifications are primarily the starting point of a development. Modeling plays a very

vital role in mimicking the system blocks (with both analog/RF and digital blocks) and influencing the parameters accurately. To go by the trend, it is desired to decrease the time to market for any new development. This means reducing the time between the drawing board design(modeling) to the actual product. To accomplish this ease of development, tools such as Matlab and Simulink which are programmer friendly (they do not involve programming complexities) are very handy. But it is observed that using such high level tools comes with a price. The final output is not highly optimum and the flexibility in-terms of modeling is limited. This was observed in the work carried out in [11]. To overcome this problem, a design approach using SystemC and SystemC-AMS would be used in this thesis. SystemC is set of C++ classes and methods that provides means to describe a digital systems from abstract specifications to register transfer level (RTL). This can be primarily used in digital system development more like a functional programming language such as VHDL. But the programming constructs are much more flexible as it is just a C++ library and the development is carried out in C++. To improve the scope of SystemC and to facilitate a complete system design (similar to a system-level tools such as Simulink), SystemC-AMS was introduced. SystemC-AMS allows hardware/software co-design at a high level of abstraction [22] allowing a wide range of usability of these libraries. This feature was a motivating factor to use this tool to model the simulators using SystemC and SystemC-AMS. The structure of the libraries used to develop this simulator is shown in Figure 4.2. From the figure, it can be observed that there is a third stack of library called TUV_AMS_Library, a library of configurable general purpose SystemC-AMS modules which was developed at the Vienna University of Technology [23].

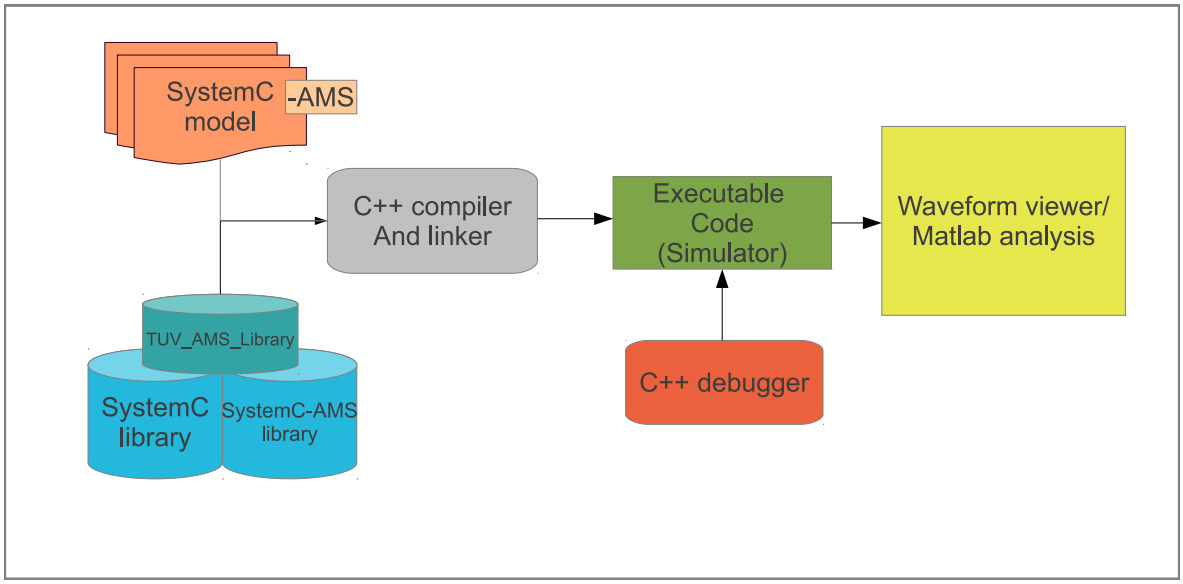


Figure 4.2: Structure of the SystemC, SystemC-AMS and the TUV libraries

SystemC-AMS provides three types of extensions:

- **Linear signal flow (LSF):** LSF can be used to model high level systems such

as adders, integrators, or transfer functions. They require a linear differential equation solver.

- **Electrical linear networks (ELN):** ELN is used to defined linear electrical network primitives such as resistors or capacitors. They also require a linear differential equation solver.

- **Timed data flow (TDF):** TDF is used in modeling high level discrete signal processing modules. The data flow simulations are performed by statically scheduling the modules before the start of the simulation.

In the simulators presenting in this thesis, all the blocks were modeled as TDF modules as it was felt that, the LSF and ELN provides higher levels of abstraction and that level of modeling was not necessary for this system as this framework was more inclined to aid in digital processing blocks. The following listing is an example of the language construct of a TDF module:

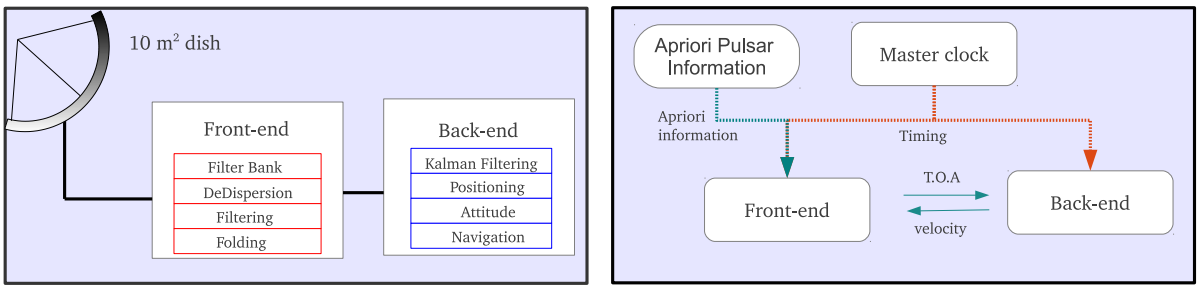
Listing 4.1: Language constructs of a generic SystemC AMS TDF module

```
SCA_TDF_MODULE(my_tdf_module) // Name of the class
{
// port declarations : the TDF module can have multiple input and output
    ports
sca_tdf::sca_in<double> in;
sca_tdf::sca_out<double> out;
SCA_CTOR(my_tdf_module) {} //Constructor
void set_attributes()
{
// module and port attributes
}
void initialize()
{
// initial values of ports with a delay
}
void processing()
{
// time-domain signal processing behavior or algorithm
}
void ac_processing()
{
// small-signal frequency-domain behavior
}
};
class my_second_module : public sca_tdf::sca_module
{
public:
// port declarations
// ...
my_second_module( sc_core::sc_module_name ) {}
// definition of the TDF member functions as done above
// ...
};
```

For more details on the language constructs of SystemC-AMS, the SystemC-AMS extensions User's Guide can be refereed[24].

4.2 Modeling a Pulsar Signal Processing System

The objective of this thesis work is to develop a framework to recreate the environment of a receiver with an antenna of $10m^2$ aperture trying to capture signals from a pulsar. Modeling such a system from the top level can be considered as a traditional communication system, with a transmitter sending unique information, a channel which corrupts this information and a receiver which compensates for the effects of the channel and extract the original information to the best possible extent (which defines the performance of the receiver). Hence a similar approach was considered in modeling this framework. This system accepts a digital form of the received signal as input and processes this raw data to extract pulsar profiles. This system is completely controlled by the back-end system which updates the processing parameters at regular time intervals. More about the dual stage processing system can be found in [9] [21] and the following Figure 4.3 is a representation of such a system.



(a) System Architecture (b) System Interfacing

Figure 4.3: System Illustration [5]

In the design of this system, only the front-end model is considered and in the following sections of this chapter, the individual blocks that were developed based on the discussion in Chapter 3 will be discussed.

4.3 System Model using SystemC-AMS

As mentioned above, the complete system was conceived based on the model of a communication system shown in Figure 4.4. This model consists of a transmitter, in our case a pulsar, a communication channel analogous to the inter stellar medium and the receiver, which will be the pulsar signal processing front-end. The implementation of each of these individual components in SystemC and SystemC AMS will now be discussed in the following sections.

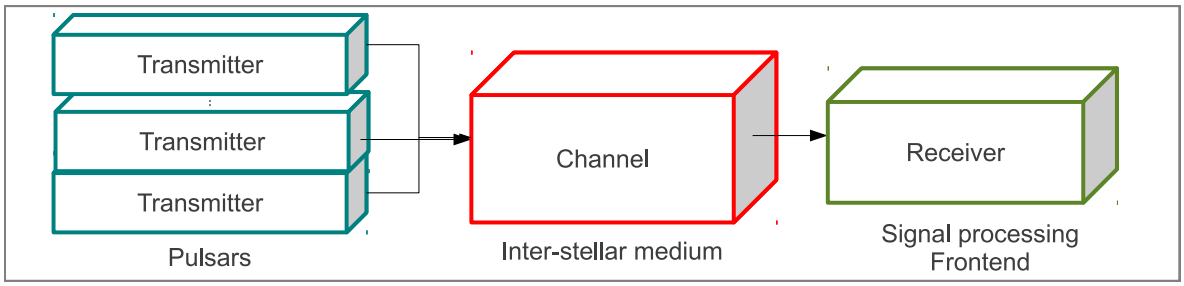


Figure 4.4: A model of generic communication system that was followed to model signals from pulsars

4.3.1 Transmitter

In the models developed from chapter 3, a transmitter can be considered as a cyclo-stationary process which repeats the signal statistics at a time period equal to the pulsar repetition period. Another aspect that was discussed was, this signal being a wide-band signal can be modeled as a white process with spectral shaping. In other words, the transmitter can be modeled as an amplitude modulated noise. Hence, to implement this module in SystemC-AMS, the following individual design blocks were developed:

- Noise shaping block.
- Noise generator, whose output levels can be controlled.

4.3.1.1 Read EPN:

The Noise shaping block, shapes the power levels of the noise generator based on the shape of the pulsar profile read from an EPN file. The EPN data format is a portable way to store folded profiles and single pulses. It is encoded in plain text, with profiles stored as hexadecimal numbers and can be used as pulsar templates. A set of EPN files containing different pulsar profiles are used to generate pulsar data. The structure of this file can be seen in Appendix A. A SystemC-AMS block is created to read the contents of this file and cyclically send the profile values at a data-rate as extracted from the EPN file (the EPN file contains information about the sample time between samples).

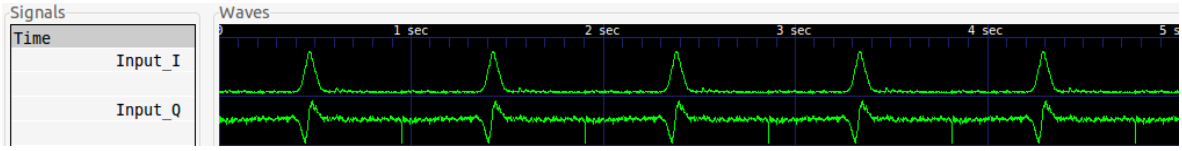


Figure 4.5: The I and Q channel output from an EPN file used to shape the noise

4.3.1.2 Amplitude Modulated Noise:

To replicate the signals from pulsars, the Tx (transmitter block) reads the .epn file, which contains the templates of known pulsars (obtained after processing by observatories) and transmits the profile data periodically with the given sampling rate (which is also obtained from the epn file). The Figure 4.5 shows the output of the transmitter module before modulating it with RF signal. The Input_I and Input_Q signals shown in the figure are read from the epn file and are transmitted via Tx ports in a loop till the desired time.

Algorithm 1: Algorithm to read the epn file and create a pulsar profile

Input: epnfile
Output: out_real, out_imag
attributes() Set the port rate of the output port as obtained from the epnfile *R*;
initialization() Read the Header of the epnfile;
Read the I, Q, U and V profile values from the epnfile;
Create the profile signal *P(t)* as a template ;
processing() Output the profile value *P(t)* at the rate R on the output ports.

Algorithm 2: Algorithm to create an amplitude modulated wide band pulsar signal

Input: in_r, in_i
Output: out_real, out_imag
attributes() Set the port rate of the output port as Fs which is given by twice the Bandwidth;
processing() Accept the input from epn profile creator described in Algorithm 1;
Create a random number between the 0 and the token accepted (P(t));
Multiply this random number with the token to create an amplitude modulated P(t);
Write the value to the output ports at the sampling frequency Fs;

4.3.1.3 Multiple Pulsars:

The transmitter block can be configured to send more than one pulsar data as shown in the figure. This feature was added so that as a simulator, it would replicate a real life condition where more than one pulsar is available in a given field of view (F.O.V) Here a selected set of 10 templates are used. The transmitter can simultaneously send 1 to 10 pulsar data depending the pre-compile configuration (this feature cannot be changed at runtime). This functionality is implemented through the use of template functionality in C++ 4.2.

Listing 4.2: Invoking multiple transmitters

```
 template <int Num_tx>
SC_MODULE(TX)
{
    //ports
    sca_tdf::sca_out<double> RF_Out[Num_tx];
```

```
        sca_tdf::sca_out<double> RF_Out_Imag[Num_tx];
        //signals
8       .
        .
};
```

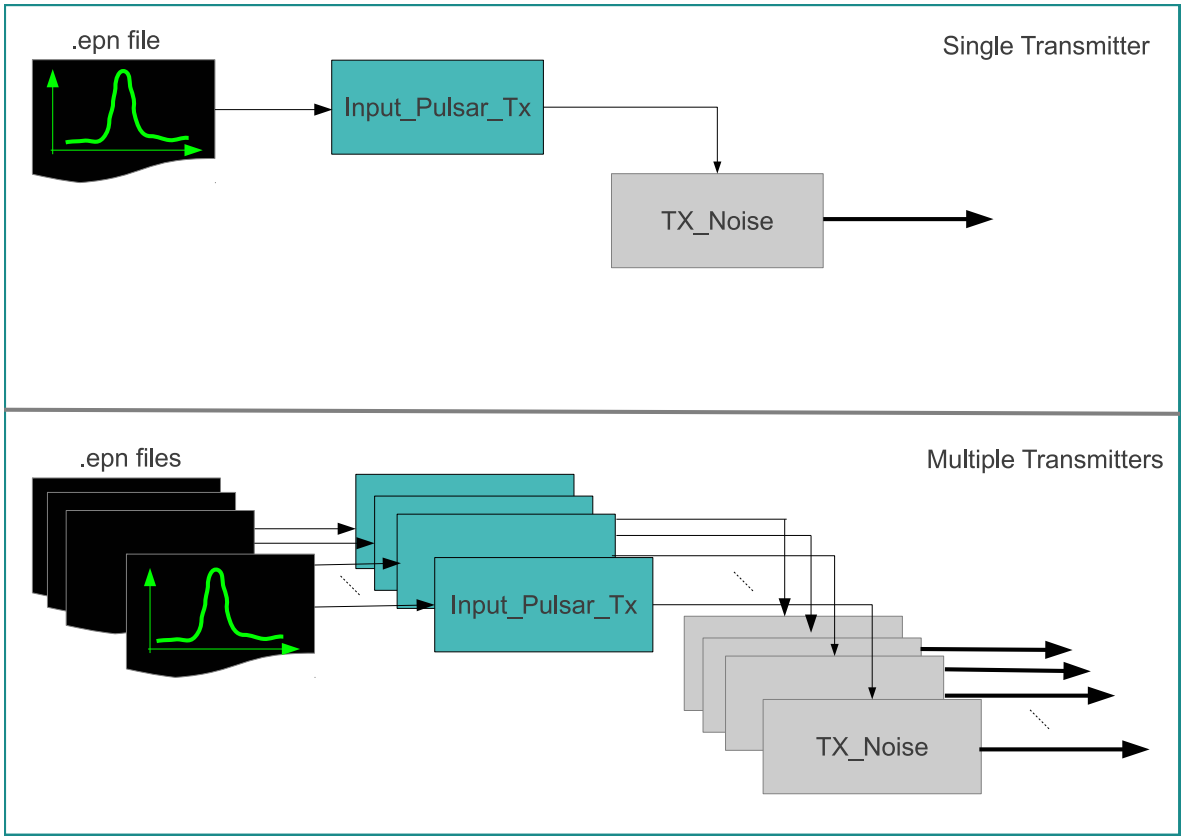


Figure 4.6: Single and multiple transmitters

4.3.1.4 Reading raw Pulsar data

A SystemC-AMS block was created to read the raw pulsar data from files. To facilitate this, two modules were written. The first module was written to read the raw data, which is the output of one of the filter banks. The second module was written to read the raw data from WSRT which is stored in the .dada format. Details regarding the contents of the first set of raw data can be accessed from a .inf file whose contents are as shown below 4.1. The second set of raw data (stored in the .dada format) consists of a header of size 4096 bytes. In this header, the details of the raw data is stored 4.2.

This raw data reading block reads the raw data from the file one by one and outputs it on its ports at a data rate obtained from the .inf file. In this case the data rate mentioned is the file is 0.0013107200 sec.

Data file name without suffix	B0329+54_081105_tracking_tile6
Telescope used	GBT
Instrument used	STELLA
Object being observed	B0329+54
J2000 Right Ascension (hh:mm:ss.ssss)	03:29:00
J2000 Declination (dd:mm:ss.ssss)	54:00:00
Data observed by	Mr. LOFAR
Epoch of observation (MJD)	54774.0
Barycentered? (1=yes, 0=no)	1
Number of bins in the time series	141312
Width of each time series bin (sec)	0.0013107200
Any breaks in the data? (1=yes, 0=no)	0
Type of observation (EM band)	Radio
Beam diameter (arcsec)	999
Dispersion measure (cm-3 pc)	0
Central freq of low channel (Mhz)	139.0625000
Total bandwidth (Mhz)	8.984375
Number of channels	248
Channel bandwidth (Mhz)	0.1953125
Data analyzed by	Jason
Any additional notes	LOFAR Test data on B0329+54 after Blue Gene upgrade

Table 4.1: Contents of the .ini file containing the raw data from LOFAR

4.3.2 Channel

The communication channel, should replicate the effects of the inter stellar medium by delaying different frequency components of the pulsar signal. The model of the channel as described in Chapter 3 is modeled as individual dispersing sections acting upon the respective pulsar signals. The resulting dispersed signals are then combined before feeding the receiver Figure 4.7. To perform these operations the channel is models using the following SystemC-AMS blocks.

- Dispersion block
- Channel Adder
- AWGN

Modeling of these blocks are explained in the following sections.

4.3.2.1 Dispersion

The dispersing block is modeled as a filter which performs frequency domain multiplication of the input signal with the coefficients generated for the respective dispersion measures, observing frequency and the bandwidth. The coefficients are generated inside the SystemC-AMS block as a part of the initialize function and at run time these

DADA HDR_VERSION	1.0
HDR_SIZE	4096
DADA_VERSION	1.0
PIC_VERSION	1.0
OBS_ID	11206003
PRIMARY	unset
SECONDARY	unset
FILE_NAME	/data2/data/06Oct2012_LEAP
FILE_SIZE	800000000
FILE_NUMBER	0
UTC_START	2012-10-06-16:53:40
MJD_START	56206.703935185185185185 2161050249
OBS_OFFSET	0
OBS_OVERLAP	0
SOURCE	B1855+09RA 18:57:36.39161
DEC	+09:43:17.2322
TELESCOPE	WSRT
INSTRUMENT	PuMa2
FREQ	1406
BW	20
TSAMP	0.025
NBIT	8
NDIM	1
NPOL	2
IN_USE	1
BYTES_PER_SECOND	80000000

Table 4.2: Contents of the header of .dada file containing the raw data from WSRT

coefficients are multiplied with the input frequency domain signal. The coefficients are generated based on the equation 4.1 and the algorithm is described in Algorithm 3.

$$H(f_o + f) \quad = \quad e^{+i\left[\frac{2\pi D}{(f_o+f)f_o^2}\right]DMf^2} \tag{4.1}$$

f_o is the observing frequency.
f is the frequency bins.
DM is the dispersion measure.

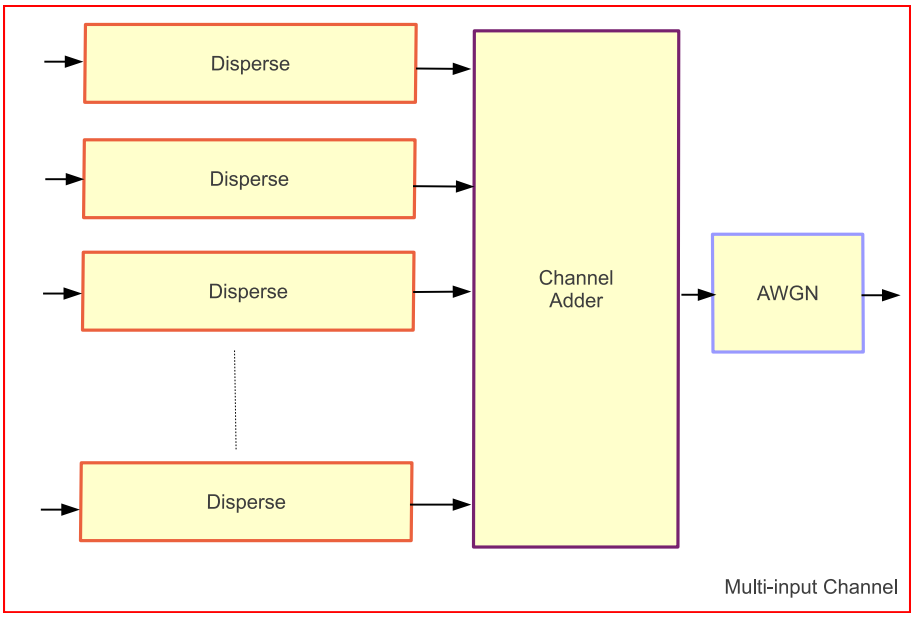


Figure 4.7: Components of the channel

Algorithm 3: Algorithm to perform dispersion

Input: in_real[FFT_Size], in_imag[FFT_Size], Dispersion_Measure, Center_frequency, Bandwidth, FFT_Size
Output: out_real[FFT_Size], out_imag[FFT_Size]
Attributes() output datarate equals input datarate;
Initialization() Frequency bins are created using the Bandwidth and FFT_Size (Bandwidth/FFT_Size);
Create an array f_h[FFT_Size] to store all the frequency values take into consideration;
Create an array to store the filter coefficients Re_coeff[FFT_Size] and Im_coeff[FFT_Size];
for *i = 0 to FFT_Size* **do**
  if *i ¡ FFT_Size/2* **then**
    Calculate the values of frequency bins and store in f_h[i];
    Calculate the real and imaginary components of the filter coefficients using the equation 4.1;
    Store the coefficients in a file for post simulation analysis;
  else
    Zero pad the filter coefficients.Re[i] = 0 and Im[i] = 0;
Processing() Accept the array of real and imaginary inputs;
Perform complex multiplication of the input array with the computed filter coefficients;
Output the product to the output ports at the same rate as input port;

The filter block accepts the inputs in parallel, hence the input data stream to the disperse block which is a serial stream of data is buffered using a serial to parallel buffer at the first stage, then the buffered data is FFTed and sent to the filter block. Only the top half of the FFTed signal is considered as the bottom half is just the reflection on the same spectrum. In order to implement this, only N/2 coefficients are generated and the rest of the bins are padded with zeros.The filtered output along with the zero padded bins are inverse Fourier transformed and serialized before sending it to the output ports using a parallel to serial buffer. This procedure is illustrated in Figure 4.8 and described in Algorithm 3.

Algorithm 4: Algorithm to perform frequency domain conversion, dispersion

Input: in_real, in_imag, Dispersion_Measure, Center_frequency, FFT_Size
Output: out_real, out_imag
Attributes() output datarate equals input datarate;
Initialization() Instantiate a serial to parallel buffer with size FFT_Size;
Instantiate an FFT engine with size FFT_Size;
Instantiate a Dispersion block (Algorithm3);
Instantiate an IFFT engine with size FFT_Size;
Instantiate a parallel to serial buffer with size FFT_Size;
Processing() Accept each of the real and imaginary input tokens and input it to the serial to parallel buffer;
for *tokens = 0 to FFT_Size* **do**
    Once the Buffer is full the serial to parallel buffer outputs the buffered tokens to the FFT engine;
    The FFTed samples are fed into the dispersing block (Algorithm3);
    Dispersed tokens are IFFTed and fed to the Parallel to serial buffer;
    Parallel to serial buffer converts the parallel input token to serial tokens;
Serial tokens are output to the output ports at the same rate as the input port.

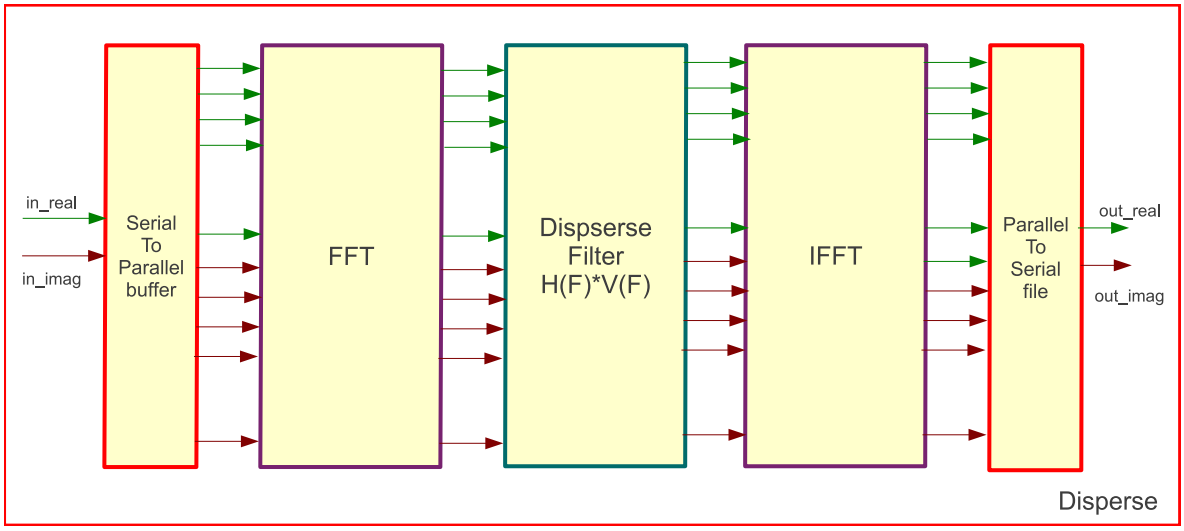


Figure 4.8: Structure of the Disperse block

4.3.2.2 Channel Adder

The pulsar signals are dispersed through individual dispersing filters and these signals have to be combined to a single signal, hence a channel adder is used to combine these signals and Figure 4.9 shows the structure of the channel adder block. It adds the individual real and imaginary components of the dispersed signals and combines them and passed to the output port.

Algorithm 5: Algorithm to integrate the individual dispersed channel components

Input: In_real[N], In_imag[N]
Output: Out_real, Out_imag
Attributes() output datarate equals input datarate;
Initialization() Create temp_real[N] and temp_imag[N] and initialize to 0;
Processing() Accept the inputs and store in temp_real[N] and temp_imag[N] respectively;
for *0 to Size of Array* **do**
  Add all the elements in the arrays;
Output the real and imaginary sum on the respective ports;

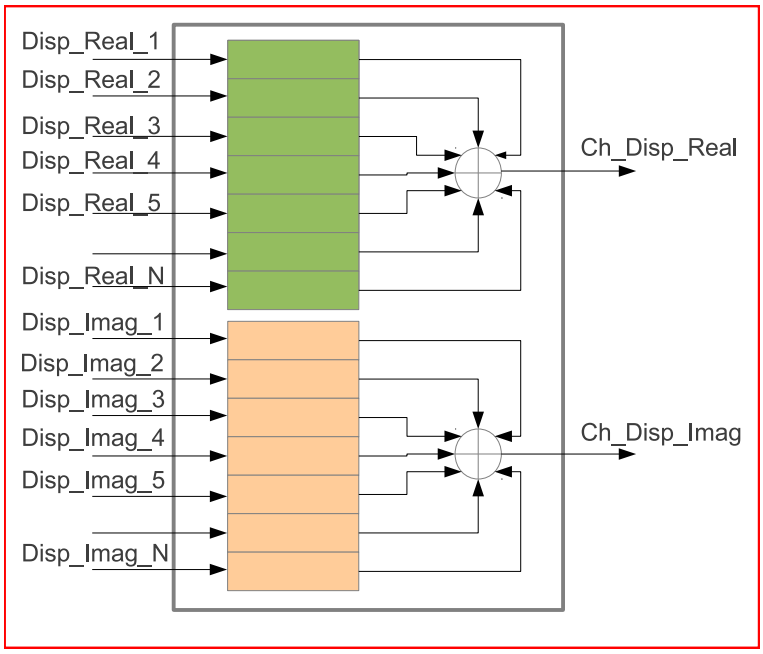


Figure 4.9: Structure of the Channel Adder block

4.3.2.3 Additive White Gaussian Noise (AWGN)

The dispersed signal from pulsars travel a considerable distance before reaching the earth and are subjected to attenuation and this effect is incorporated using the "Air" class from TUV_AMS_Library, which was modified to accommodate complex signals. The attenuation and the additive noise components are the input parameters to this SystemC-AMS module. More about this block can be obtained from [25].

4.3.3 Receiver

The focus of development in this thesis work is to design and test a signal processing receiver. From the signal modeling aspects discussed in Chapter 3 and from instrumentation section in Chapter 2 the receiving block consists of the following processing units:

- Poly-phase filter bank
- De-dispersion
- Folding and detection

In the model of the receiver considered in this framework, a directional antenna with a restricted field of view is also considered. The features of this antenna is modeled as a systemC-AMS block which accepts multiple inputs and attenuates the input signal with different attenuation factors which correspond to the pointing error. From an implementation perspective, the antenna has to be modeled as a part of the receiver, but in the channel model discussed above, the individual dispersed signals are combine. Hence if the antenna is modeled as a part of the receiver, the received signal will contain signals from all the pulsars. So the effects of the antenna is modeled inbetween the transmitter model and the channel model. As a result the signals are attenuated before the signals are passed through the dispersion filter and combined. This is shown in Figure 4.10 and the Algorithm to perform this on SystemC-AMS is described in Algorithm 6.

Algorithm 6: Algorithm to implement the restricted field of view of the antenna

```
Input: Tx_in_r[N_Tx] Tx_in_i[N_Tx]
Output: Tx_out_r[N_Tx] Tx_out_i[N_Tx]
Initialization() initialize the error angles to zero ;
Processing()  for o to Number of transmitters do
    Accept the input signals ;
    compute the error angle <= (Position of Pulsar) - (Position of Antenna) ;
    if error angle ¡ Beam width/2 then
        compute cos(α_error)cos(β_error) * input ;
    output the computed values on the output ports ;
```

4.3.3.1 Poly-phase filter bank

The functional details of a poly-phase bank is discussed Chapters 2 and 3. In this section, implementation of this filter in SystemC-AMS is provided. The basic building blocks of a polyphase filter, as discussed earlier consists of delay units, low pass filters to prevent aliasing and FFT unit. This filter is implemented in SystemC-AMS as shown in Figure 4.11. The delay unit is implemented as a serial to parallel buffer with an overlap of $N-1$ for an N channel poly-phase filter and the data rates of the input and output ports are equal. These delayed samples are low pass filtered at cut off frequency $f_c = BW/N$ to avoid aliasing of the decimated/down-sampled. The down-sampled

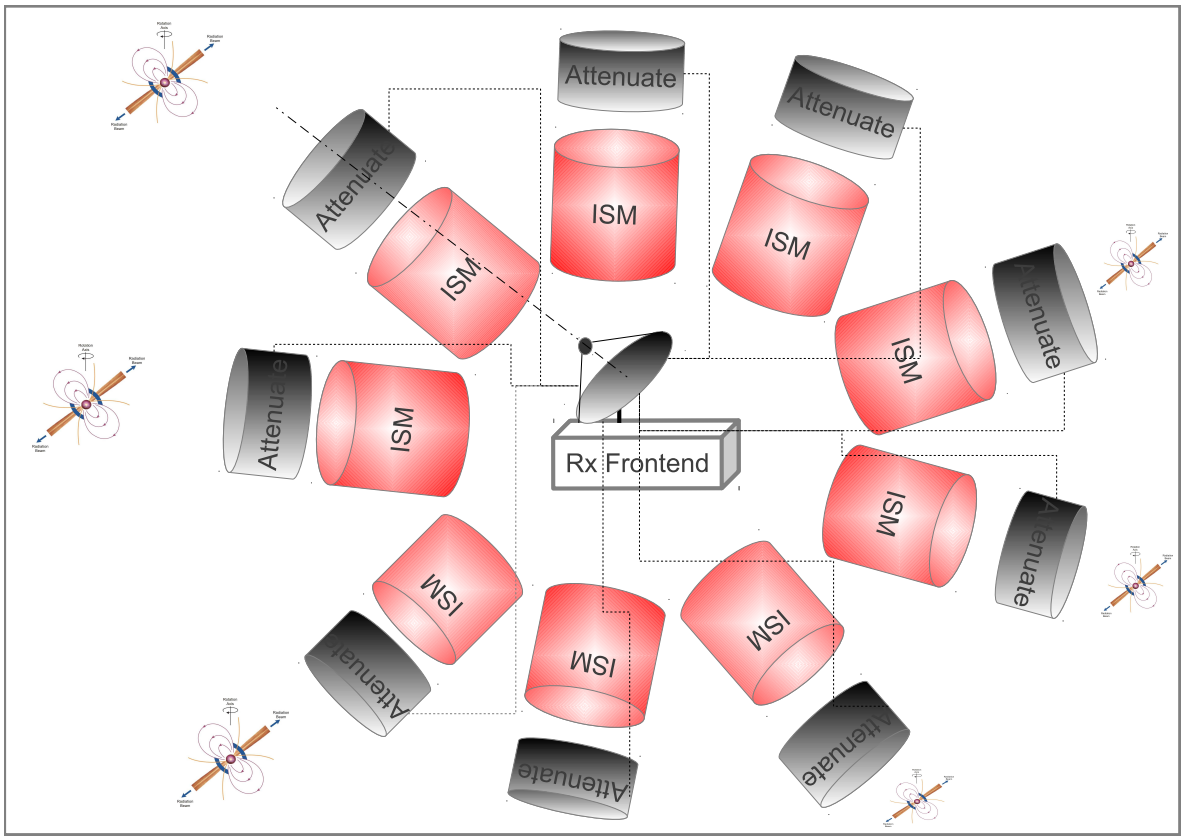


Figure 4.10: Structure of the Channel Adder block

samples are FFTed and provided on the output ports to perform further processing on smaller frequency bands.

4.3.3.2 De-disperse

The time delayed frequency components of the received signal which is first split into smaller sub-bands using the poly-phase filter bank is used for further processing to compensate for the delays. This is done by passing the subbands through a de-dispersion filter which converts the incoming signal to frequency domain and then multiplies it with the coefficients of the $H^{-1}(F)T(F)$. The coefficients are generated during initialization of the SystemC-AMS block and during the run time, complex multiplication is performed between the coefficients and the incoming real and imaginary data samples. In the block attributes, the data-rate of the ports of this block is set to default value 1. Hence this blocks outputs the processed data at the same rate at which data is input to this block. Parameterizable sections of this block is the number of point FFT to be performed on the input data stream. This is passed as a template parameter which defines the size of the input buffer, size of the FFT and size of the coefficients to be generated.

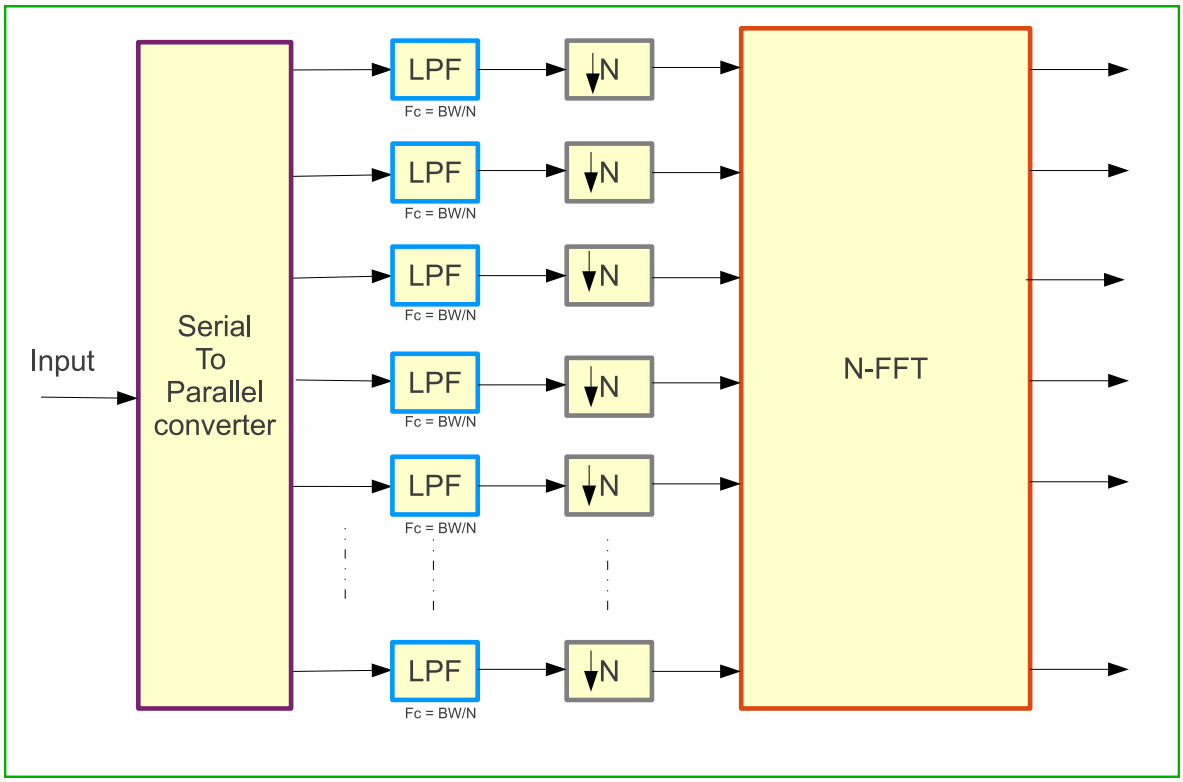


Figure 4.11: Structure of the SystemC-AMS Polyphase filter bank

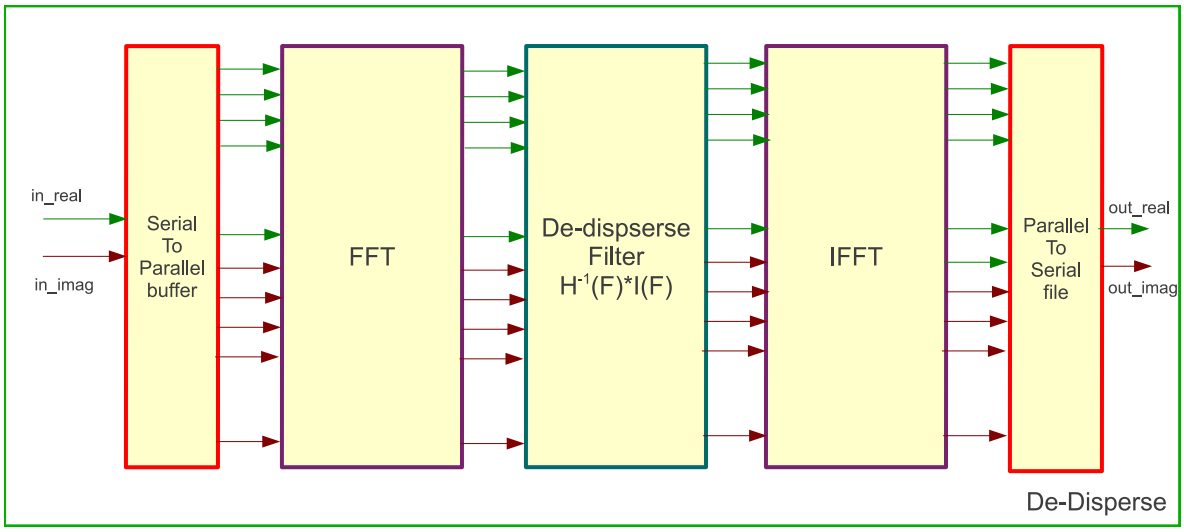


Figure 4.12: SystemC AMS block to perform de-dispersion.

4.3.3.3 Folding

The process of folding as illustrated in the Figure 4.13 can be imagined as circular stack of memory banks which are initialized with zeros and as and when the data processing starts, the received data samples are filled into the circular memory blocks. Once the

last bin is reached the index is set back to the first location and the received data sample is added to the existing data sample. As the poly-phase filter bank channelizes the input data stream into smaller frequency bands, folding has to be performed on each of these sub-bands. This operation of folding multiple frequency bins can be imagined as a large stack of circular memory as shown in Figure 4.13.
This folding operation was conceptualized in two ways, the first method was a simple non-adaptive folder which is suited for folding of signals whose periods are known and the number of bins or memory locations in the circular path is a variable but fixed during compilation. The second folder is an adaptive folder which transforms the input raw signal to its respective phase and stored in the circular format.

The algorithm for a simple folder is shown in algorithm 7:

Algorithm 7: Algorithm to perform folding

Input: bins, offset, fp_fold, in_r and in_i
Output: Fold output stored as a file
Initialization() create a file pointer using the fp_fold parameter;
create a temp array of size bins for storing real and imaginary inputs;
Processing() Accept the real and imaginary input samples;
Calculate the index value to store the respective sample in the array;
Add the received sample to the value from the array of the calculated index;
for *0 to bin_size* **do**
  Update the output file with $\sqrt{(real^2 + imag^2)}/Fold\_Counter$;
if *index is less than bins* **then**
  increment index;
else
  initialize index to zero;

4.4 Introduction to SynRaPSor and RaPSor

Using the individual signal processing blocks discussed in the previous section, two simulators were developed. The first simulator named as **Syn**thesized **Ra**w **P**ulsar **S**ignal generat**or** and Process**or (SynRaPsor)** is used to generate synthetic raw pulsar data and using this data the receiver is tested. All the blocks used in this simulator are SystemC-AMS based blocks, hence the receiver under test cannot be directly synthesized to actual hardware, but the complete functionalities of this receiver can be tested under different configuration modes. The second simulator that was developed was named as **Ra**w **P**ulsar Process**or RaPSor**. As the name suggests, this simulator is used to test the receiver with raw data from the RF front-end setup. Again, this simulator has been provided with configurable features so that it can be used to test in different configuration modes. The development of a simulator with synthesisble receiver blocks was also carried out but was not completed in the given time-frame of this thesis, hence the details of these blocks are discussed in the Appendix B. Figure 4.14 (provided at the end of this chapter) gives a classification of the different simulators.

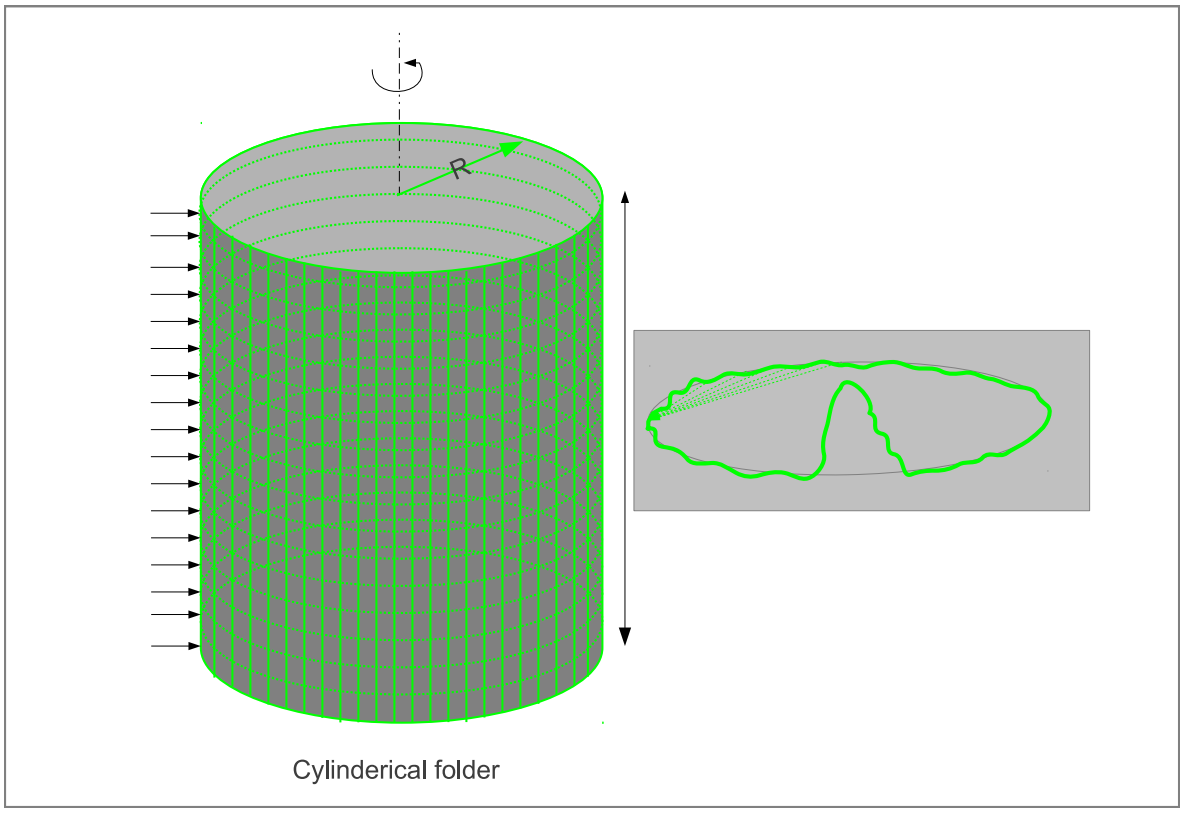


Figure 4.13: SystemC AMS block to folding.

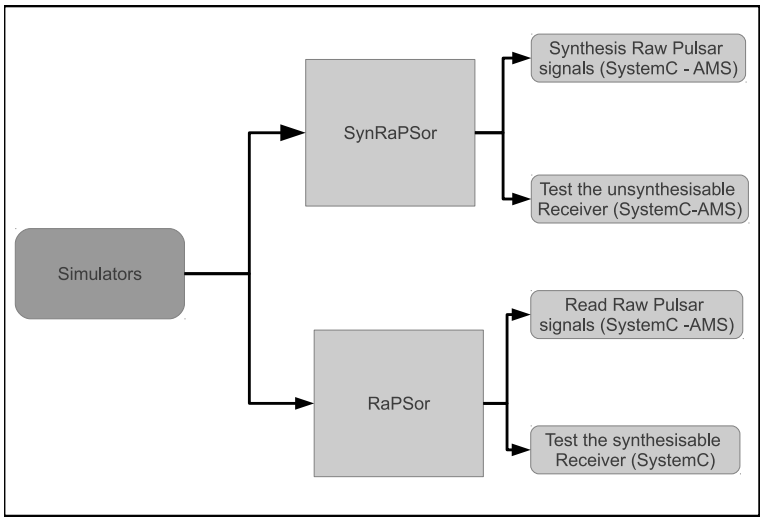


Figure 4.14: Overview of the Simulators developed

4.5 SynRaPSor: Synthesized Raw Pulsar Signal generator and Processor : Simulator overview

SynRaPSor is a SystemC-AMS simulator which generates pulsar signals, induces channels effects and performs pulsar signal processing operations to retrieve the signal. The complete architecture of this simulator is shown in Figure 4.15 (provided at the end of this chapter). From the figure, it can be observed that the simulator can be configured to have more than one transmitter, which in real life corresponds to the real life scenario where the received signal could be from more than one pulsar. If there are many pulsars along the line of sight, to lock to one of the multiple pulsars, the received signal has to be processed with the respective dispersion measures and folding periods. It is also seen that the each of these blocks i.e. the transmitters, channels and the receiver are controlled through a common system controller which is used to configure them with the desired numbers as shown in Listing 4.3.

The individual signal processing blocks are stored as libraries. These libraries are instantiated in the main simulator as standard header files. This is illustrated in Listing 4.3.

Listing 4.3: Header files containing the

```
#include "systemc-ams.h" //SystemC AMS Library
#include "tuv_ams_library/tuv_ams_library.h" // TUV AMS library for FFT
    blocks
#include "EPN_File_read.h" //To read the .enp files
#include "Pulsar_transmitter.h" //To instantiate the transmitter blocks
#include "Ch_lib.h" //To instantiate the Channel blocks
#include "Rx_lib.h" //To instantiate the Receiver clocks
```

4.6 RaPSor: Raw Pulsar Processor

This simulator is designed to test the receiver module discussed in SynRaPSor using the raw data obtained from the RF-frontend. This simulator can be used to test the validated design of the receiver from SynRaPSor to test with real pulsar data. This simulator would be very useful to process the raw data from the hardware setup to detect the pulsars and the output of this simulator is displayed as a graph while the data is being processed. This feature helps in checking whether there is any drift in the frequency and correspondingly the dispersion measures can be changed. Another feature which was implemented as a part of this simulator is that parts of this simulator can be enabled or disabled based on the raw data being processed. If the raw data is from the RF-front-end setup in EWI, then it needs to first be processed to get smaller frequency bands, hence polyphase filter bank can be enabled. Similarly, if the processing bandwidth is very small, then the influence of dispersion inside this frequency band will also be very small. Hence in these cases the de-dispersion blocks can be disabled and folding can be performed directly. A diagrammatic representation of this simulator is shown in Figure 4.16.

SystemC-AMS provides a feature where the intermediate signals in the simulator can be dumped into a trace file and at the end of the simulation, this trace file (stored

Table 4.3: Configurable parameters of the Simulator

Parameter	Data type	Value
f_o	double	1200
BW	double	10
Fs	double	2e6BW
Fp	double	2000
D	double	4.148808e15
Channels	const int	256
N_tx	const int	1
Overlap_tx	const int	0
file_name[0]	char*	./Pulsars/B0031-07.410.epn
file_name[1]	char*	./Pulsars/B1338-62.1411.epn
file_name[2]	char*	./Pulsars/B0052+51.1408.epn
file_name[3]	char*	./Pulsars/B0031-07.1408.epn
file_name[4]	char*	./Pulsars/B0031-07.1560.epn
file_name[5]	char*	./Pulsars/B0052+51.925.epn
file_name[6]	char*	./Pulsars/B0031-07.606.epn
file_name[7]	char*	./Pulsars/B1112-60.1440.epn
file_name[9]	char*	./Pulsars/B1820-14.606.epn
file_name[8]	char*	./Pulsars/B0031-07.925.epn
Tx_Azi[10]	double	0, 10, 60, 80, 90, 120, 160, 210, 240, 300
Tx_Ele[10]	double	0, 10, 30, 30, 60, 70, 90, -10, -45, -80
N_ch	const int	1024*2
Overlap_ch	const int	0
DM_ch[10]	double	10,0,0,400,500,600,700,800,900,100
atten_ch	double	0.6735
n_va_ch	double	0.5
Rx_Azi	double	0
Rx_Ele	double	0
beamwidth	double	60
DM_Rx	double	10
N_rx	const int	16
Bin_Size	const int	447

in vcd format) can be viewed using open-source tools such as GTK-wave. But in this simulator, it was intended to view the data real-time as it is being processed to give a visual understanding of the processing. Hence, in this simulator, the result of the folding block is passed to a plotting tool(Matlab) via pipes and the data is updated on the plot in realtime. At the end of the simulation, the temporary pipe is removed. Listing 4.4 shows how this feature was implemented in SystemC-AMS.

Listing 4.4: Passing the intermediate signals to plotting tools via pipes

```
system("mkfifo /home/vigneswaran/workspace/RaPSor/result1.dat");
system("/usr/local/Matlab/R2012a/bin/matlab -nodesktop -nosplash < /home
```

```
    /vigneswaran/workspace/RaPSor/readpipe.m &");
 cout<<"signal tracing start"<<endl;
 sca_util::sca_trace_file* atf = sca_util::sca_create_tabular_trace_file(
    "/home/vigneswaran/workspace/RaPSor/result1.dat");
 for(int i=0;i<bins;i++)
{
    sprintf(trace_Fold_signal_Real[i],"Fold_R%d",i);
    sca_util::sca_trace( atf,Folder_Real[i], trace_Fold_signal_Real[i]);
    sprintf(trace_Fold_signal_Imag[i],"Fold_I%d",i);
    sca_util::sca_trace( atf,Folder_Imag[i], trace_Fold_signal_Imag[i]);
}
cout<<"Simulation started"<<endl;
sc_start(Sim_time,SC_SEC);
cout<<"Simulation ended"<<endl;
sca_util::sca_close_vcd_trace_file( atf );
system("rm /home/vigneswaran/workspace/RaPSor/result1.dat");
```

4.7 Summary

Two simulators were developed using SystemC-AMS and SystemC C++ libraries. These simulators are first of their kind to make use of standard SystemC-AMS libraries to model a mixed signal system with the analog signals being signals from celestial objects. The simulators were developed to accommodate features such as multiple transmitters, multiple channels and receiver with restricted field of view emulating the features of using a dish antenna and processing blocks with different configurations. The first simulator SynRaPSor was can be used to generate wide-band signals from pulsars and various dispersion measures. The receiver block is also integrated in this simulator to enable testing and validating the receiver blocks. The second simulator RaPSor was developed to test the receiver with the raw pulsar data obtained from a real hardware. The present simulator was tested with a dataset provided by LOFAR and WSRT. The results of these tests are provided in the following chapter.

Figure 4.15: SynRaPSor Simulator Overview

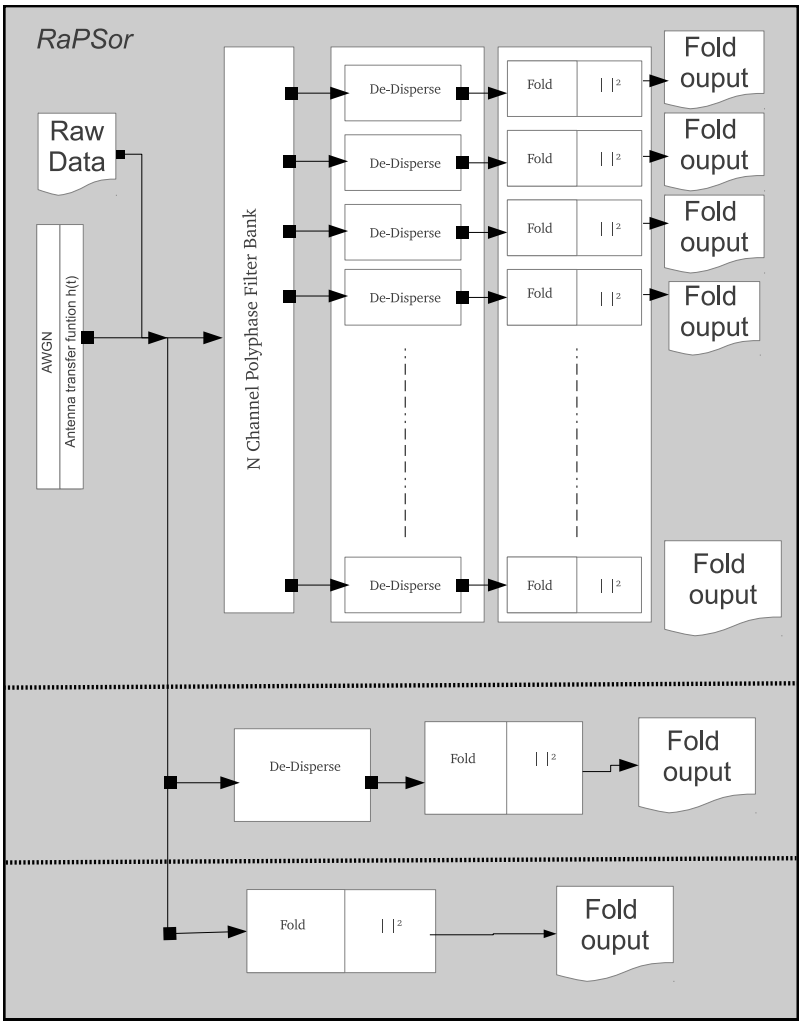


Figure 4.16: RaPSor Simulator Overview

Results, Analysis and Road-map for Future Development

5

In this chapter, all the results obtained in this thesis work are discussed. First, the results of individual SystemC-AMS blocks are provided. Following this, the results of the individual simulators SynRaPSor and RaPSor are discussed. To conclude with, a road-map which would give a clear direction for any future development work to be carried out in the development of this navigation system is presented.

5.1 Results

SystemC-AMS and SystemC libraries provide an option to trace the intermediate signals by dumping the signals with a time resolution as defined in the simulator. The usual convention is to set the resolution to 0.1 *psec* which is shown in the Listing 5.1. The trace file created can be later used to view the results using open source tools such as GTK-wave. The same procedure was followed initially to view all the simulation results. But as simulation times and processing bandwidths increased, this method could no longer be used. Hence alternative methods using pipes and plotting using Matlab was followed. The intension was to make use of open source to perform the data acquisition but due to the limited time frame, and due to the ease of usability of Matlab, it was decided to use Matlab to plot all the simulation results. In the following sections the results are discussed using the plots.

Listing 5.1: Setting the time resolution of the simulator

```
sc_set_time_resolution(0.1,SC_PS);
```

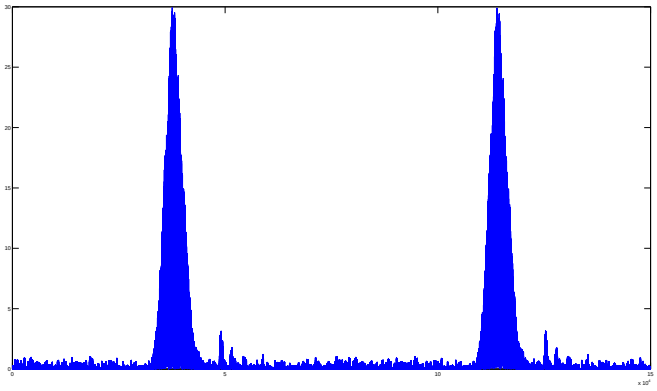
5.1.1 Individual Processing blocks

In the first section of results, the working of the individual processing blocks are discussed in the sections 5.1.1.1,5.1.1.2 and 5.1.1.3. Following this, the results of SynRaPSor are elaborated in 5.1.2 and in 5.1.3.3, the results of running raw data on RaPSor are discussed.

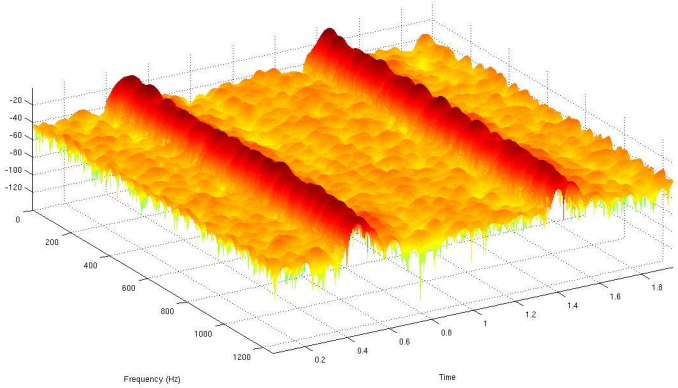
5.1.1.1 Transmitter

The transmitter module created is capable of recreating a wide-band pulsar signal. The resulting output at the port of a transmitter are shown in Figure 5.1a and 5.1b. Figure 5.1a shows the time domain representation of the pulsar signal and Figure 5.1b shows the spectrogram view of the generated pulsar signal. In the time domain representation, it can be seen that the profile of the pulsar signal varies and the signal statistics repeats itself at periodic time intervals. In the spectrogram representation of the signal, it can

be seen that the signal occupies a wide band of frequency and the pulses at different frequencies travel with the same group velocity. The shape of the pulse profile generated



(a) The output the transmitter block from a pulsar in time domain (Number of transmitter = 1)

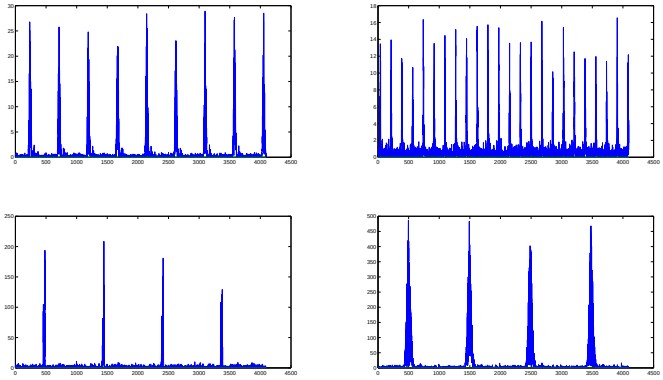


(b) The output the transmitter block from a pulsar a spectrogram view (Number of transmitter = 1)

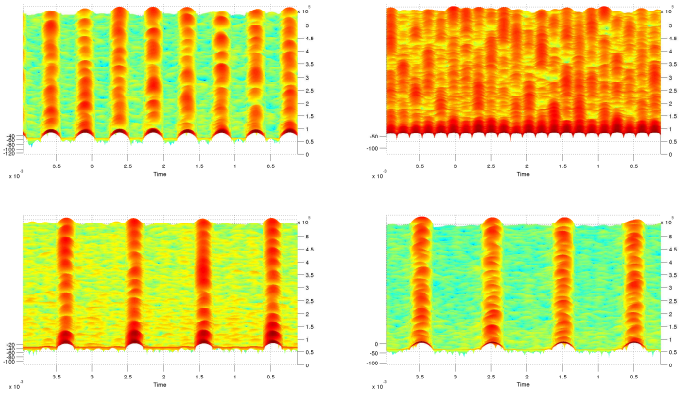
Figure 5.1: Output of RaPSor which plotted in Real-time

is modulated with the pulse profiles of an actual pulse's profiles which are stored in epn formats. So, the shape and period of this generated signal can be changed by using a different epn file. This is shown in the next figure where multiple transmitters are instantiated. It was discussed earlier that the transmitter can be configured to transmit as multiple pulsars at the same time. In the Figure 5.2 an example with four instantiations is shown in time domain as well as spectrogram view. It can be seen that there are four pulsar signals of different pulsar periods simultaneously being used.

Having successfully recreated the transmitted signal which confirms to the properties as mentioned in the literature lets next have a look at the modeling results of the channel.



(a) The output the transmitter block when configure to send multiple pulsar signals (Number of transmitter = 4)



(b) The output the transmitter block when configure to send multiple pulsar signals in a spectrogram view (Number of transmitter = 4)

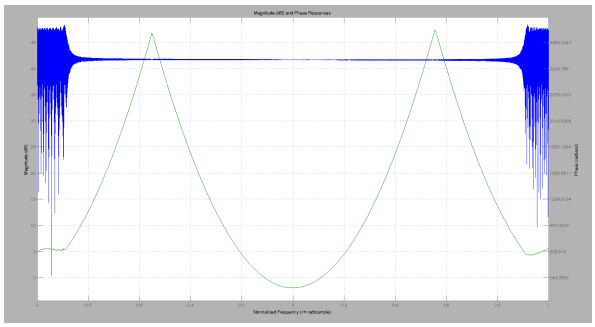
Figure 5.2: Multi transmitter configuration

5.1.1.2 Channel

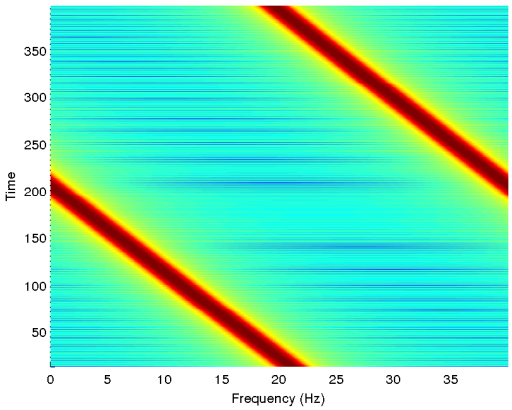
In this section, the results of the channel blocks will be discussed. From the discussions in the earlier chapters, it is evident that the communication channel is a combination of a phase only filter and an Additive White Gaussian Noise (AWGN). Hence in this section, let us have a look at the various aspects of the filter using the generated filter coefficients and the effect of this filter on pulsar signals.

The channel model was implemented by creating a phase only filter $H(f + f_o)$ whose spectrogram view and the magnitude and phase response views are shown in Figure 5.3. It can be seen that the phase varies with the frequency and in the spectrogram view it is seen that the lower frequency components are delayed as compared to the higher frequency components.

Now, without adding any additive noise component, the output after applying the



(a) Magnitude and phase response of the $H(f+f_o)$ filter



(b) Spectrogram view of the $H(f+f_o)$ filter

Figure 5.3: Filter response of the $H(f+f_o)$ and Spectrogram view

channel effects (with different dispersion measures) to the wide-band pulsar signal is shown in Figure 5.4. It can be clearly seen that at zero dispersion measure, all the frequency components of the pulsar signal travel with the same velocity. But upon increasing the dispersion measure, the lower frequency components are delayed much more than the higher frequency components.

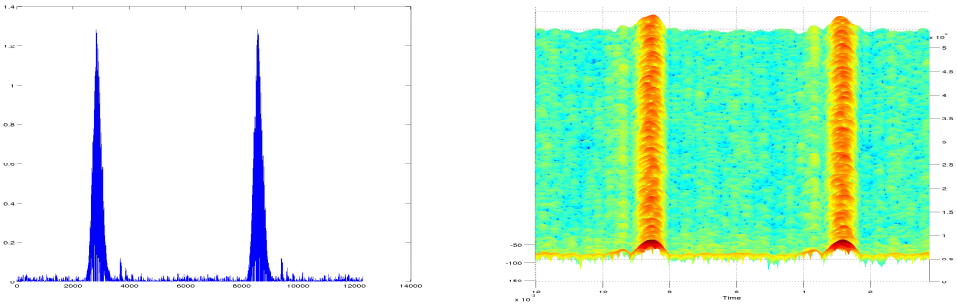
The output of the channel was also checked for AWGN. Hence after adding noise levels of -10 dB, -20 dB, -30 dB and -40 dB the respective channel outputs are shown in Figure 5.5. From the figure it can be observed that the clean pulsar signal shown in Figure 5.1 gets buried inside the noise. The Figure 5.5 shows both the time domain representation and the spectrogram view to give an idea of how the noise levels affect the clean pulsar signal.

Following the results obtained after channel effects, in the next section, the results of the receiver module are presented.

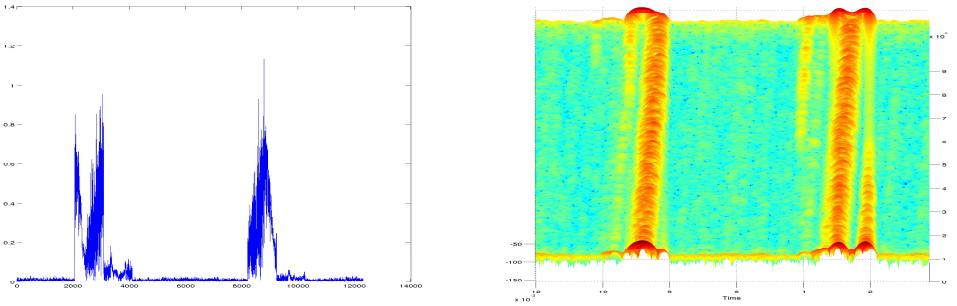
5.1.1.3 Receiver

The receiver consists of various processing blocks and in this section, the outputs of all these individual blocks are discussed. The first processing block encountered in the receiver is the polyphase filter bank and Figure 5.6a shows the impulse response of a 16 channel polyphase filter bank which was developed using SystemC-AMS. The output of this filter bank is shown in 5.6b, where the output of a 8 channel PFB capturing a signal which has undergone dispersion with a dispersion measure of 30 is shown.

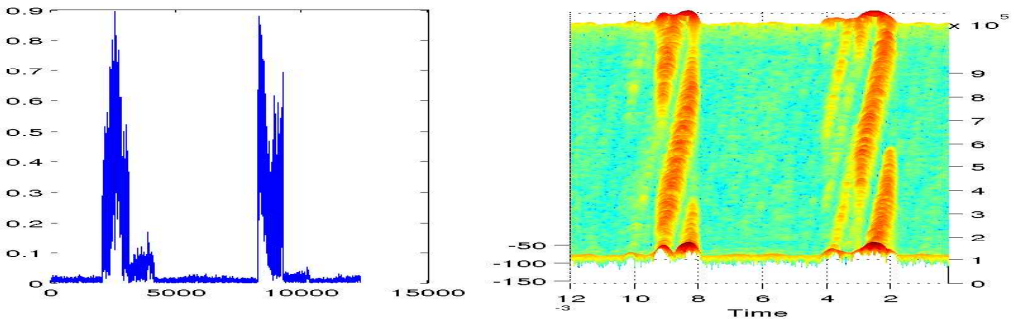
Following the filterbank, the next processing block is the de-dispersion filter which is used to reverse the effects of the inter-stellar medium. This filter is very similar to the implementation of the dispersion filter and the output of this filter is shown in Figure 5.7. In this figure, the signal in red indicates the dispersed signal and that in blue indicates dedispersed signal.



(a) Time domain and spectrogram view of the output of the channel for a dispersion measure of 0



(b) Time domain and spectrogram view of the output of the channel for a dispersion measure of 10



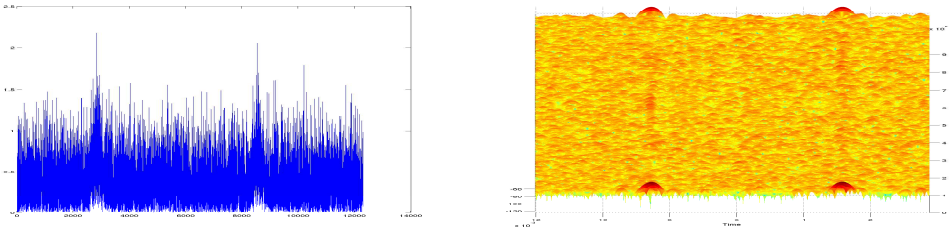
(c) Time domain and spectrogram view of the output of the channel for a dispersion measure of 30

Figure 5.4: Output of the channel without additive noise

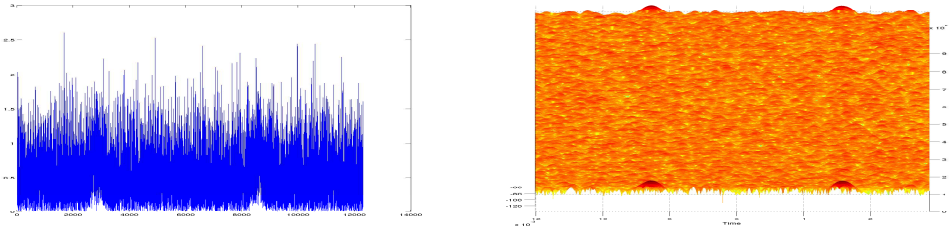
The final processing block of the receiver is the folding block, which repeatedly folds the received data over a specified time period. The output of this block is shown in Figure 5.8, which represents the folded output after removing the dispersion effects from the received signal. The figure shown is obtained after performing folding for a period of 10000 seconds.

5.1.2 SynRaPSor

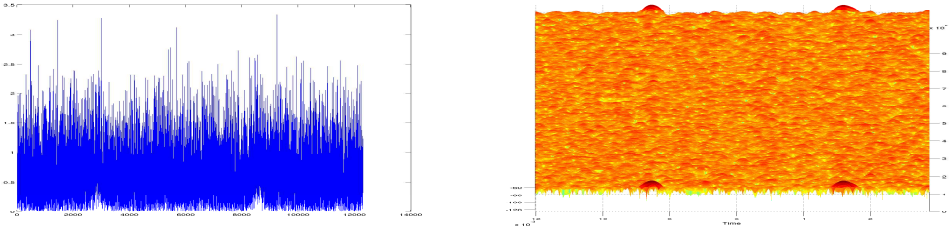
SynRaPSor, which is the complete simulator instantiates all the three primary blocks discussed above. and the result of this is shown in Figure 5.9 where the top most spec-



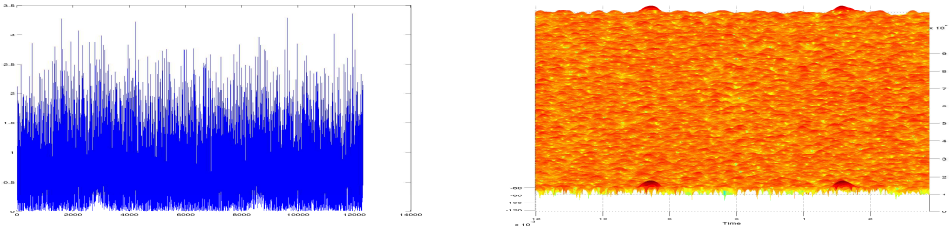
(a) Time domain and spectrogram view of the output of the channel after adding noise of -10 dB



(b) Time domain and spectrogram view of the output of the channel after adding noise of -20 dB



(c) Time domain and spectrogram view of the output of the channel after adding noise of -30 dB

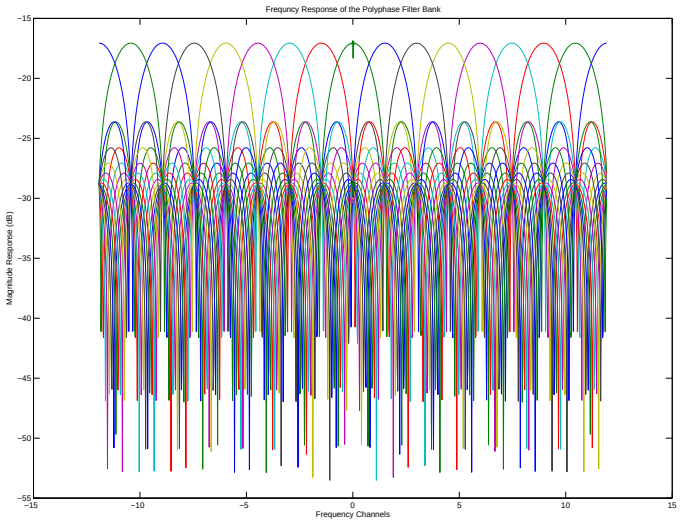


(d) Time domain and spectrogram view of the output of the channel after adding noise of -40 dB

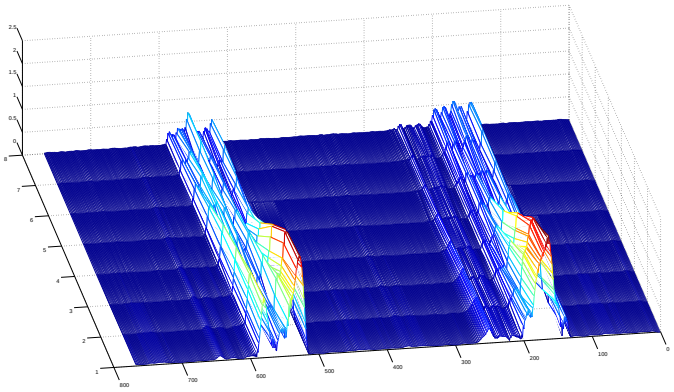
Figure 5.5: Output of the channel after adding AWGN

trogram represents the output of the transmitter, the middle spectrogram represents the output of the channel and the last figure gives the output of the receiver after de-dispersing the received signal. The simulator was subjected to various experiments by changing the configuration parameters and in the following sections these results are discussed.

The simulator was tested for comparing the result of folding the data with and without performing de-dispersion and Figure 5.10 gives the result of this experiment. It can be seen that the width of the folded signal without de-dispersion was much wider



(a) Impulse Response of a 16 channel poly-phase filter bank



(b) Output of an 8-channel polyphase filter bank with a dispersed signal of 30 at 3 KHz

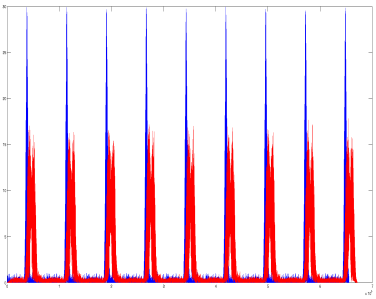
Figure 5.6: Polyphase filter bank outputs

than the actual pulsar signal in Figure 5.10a. Contrary to this, the folded output after performing de-dispersion shows that the output pulse width is comparable to the pulse width of the transmitted signal in Figure 5.10b.

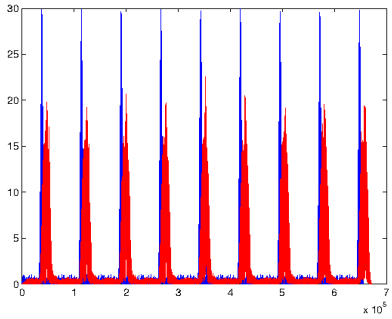
Further analysis that was performed using the SynRaPSor is described in the following section on analysis where, various parameters of the simulator were analyzed.

5.1.3 Analysis

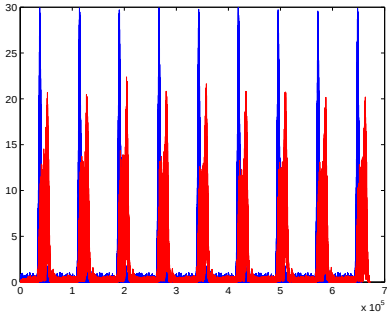
Based on the findings of the experiments carried out using the developed simulator the following analysis was carried out:



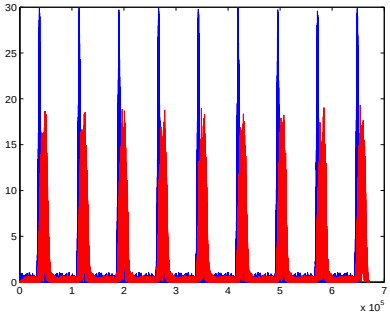
(a) Dispersion measure = 10, Bandwidth = 20 MHz, Observing frequency = 1200 MHz



(b) Dispersion measure = 20, Bandwidth = 20 MHz, Observing frequency = 1200 MHz



(c) Dispersion measure = 30, Bandwidth = 20 MHz, Observing frequency = 1200 MHz



(d) Dispersion measure = 40, Bandwidth = 20 MHz, Observing frequency = 1200 MHz

Figure 5.7: Time domain representation of the effect of De-dispersion filter (Signal in red indicates dispersed signal, signal in blue indicates dedispersed output)

5.1.3.1 Simulator Analysis

The limits of the simulator were tested by increasing the size of the FFT in the channel model. This was directly related to the achievable generated bandwidth of the pulsar signal. During this analysis it was observed that the execution time of the simulator increased exponentially as the size of the FFT was increased. Beyond an FFT size of 2^15 the simulator failed to perform as the scheduler failed to create the schedule list. Hence it was concluded that the maximum size of the FFT supported by the simulator was 2^15 point FFT. This is shown in Figure 5.11, where it can be seen that the execution time increases drastically with the increase in FFT size.

The simulator was tested for the achievable output SNR levels for various input noise levels (such as -20 dB, -30 dB, -40 dB and -50 dB) verses simulation time for which the signals are processed. In this experiment, a dispersion measure of 30 was considered (bandwidth of 3 kHz at 1200 MHz observing frequency). It was observed that as the simulation time was increased, there was an increase in the output SNR. But this rate of increase was higher for lower input noise levels such as -20 dB but

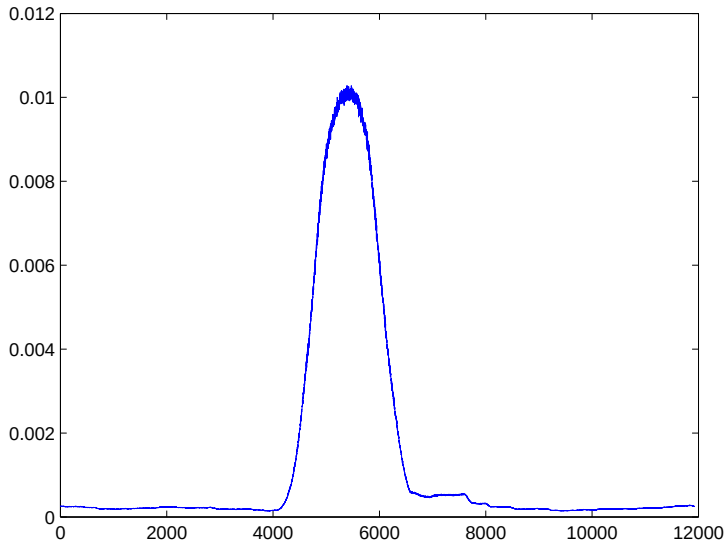


Figure 5.8: The output of the fold module after folding for a period of 10000s.

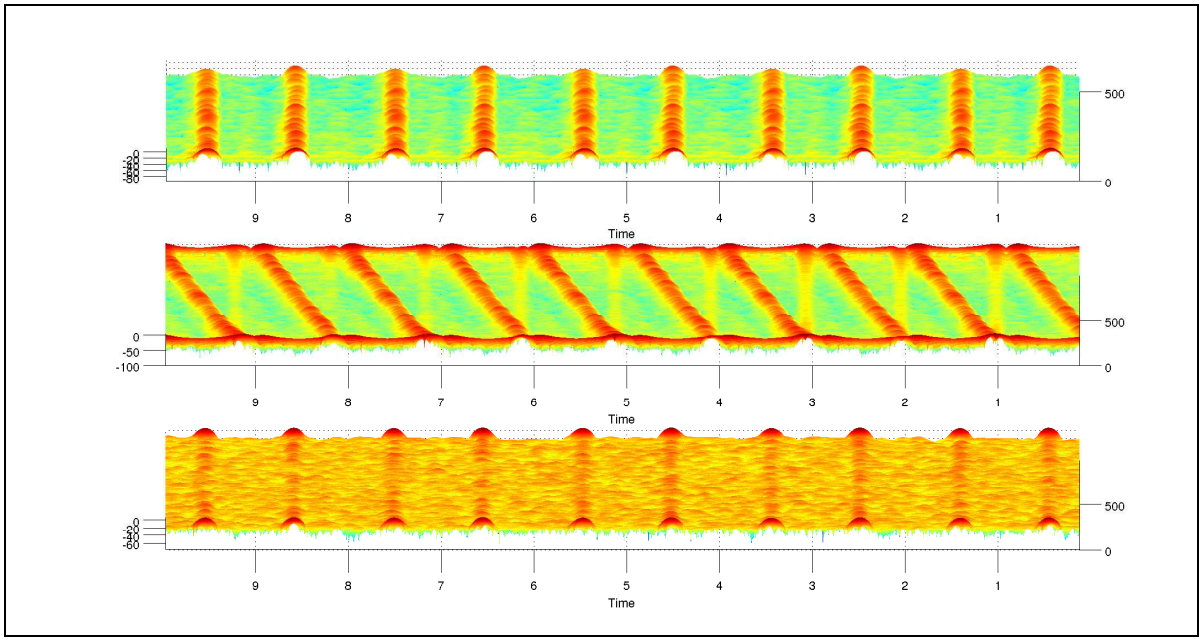
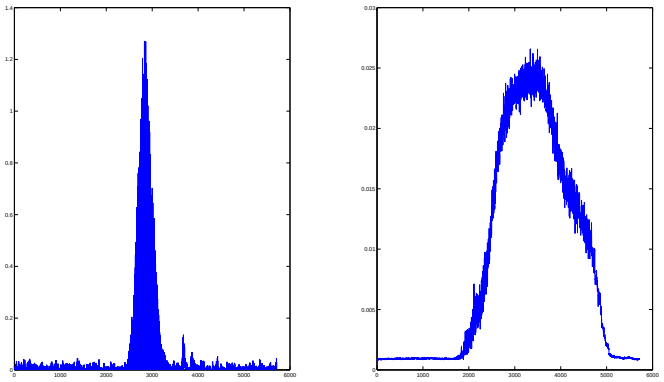
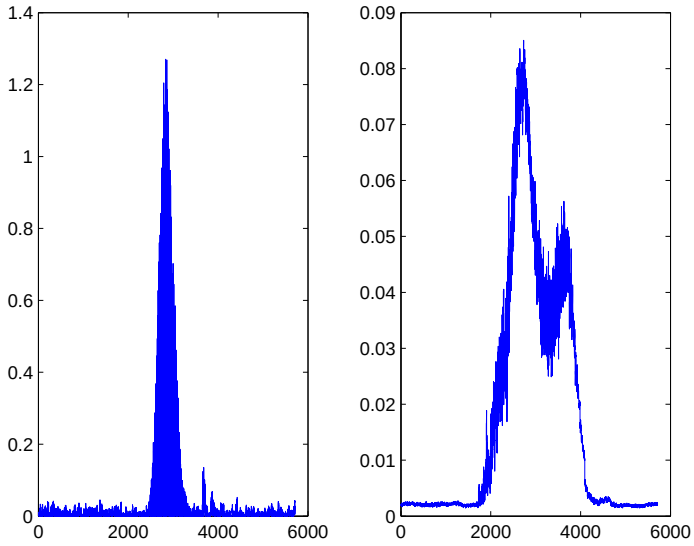


Figure 5.9: The output of the transmitter, dispersive channel and the de-dispersed signal be the receiver.

as the channel noise levels increased, this had little effect on the simulation time as observed for -50 dB.



(a) Folded output without de-dispersion



(b) Folded output with the de-dispersion filter

Figure 5.10: Comparison between folded outputs with and without de-dispersion filter

Another very interesting parameter to be analyzed was the output SNR verses the dispersion measure in the channel. As the dispersion measures were increased, the output SNRs was observed to decrease as expected. Considering that this result is taken after an integration time of 1000 s, it is important to know to what extent is it advisable to allow the system to perform de-dispersion to view the received signal.

The receiver consists of a polyphase filter bank which is used to channelize the income data stream. Hence, experiments were conducted by increasing the size of the polyphase filter bank. The result of this experiment is shown in Figure 5.14. From the result, it was observed that the simulator reaches its limit when the size of the polyphase filter bank reaches 64 channels. Beyond this number, the simulator fails at

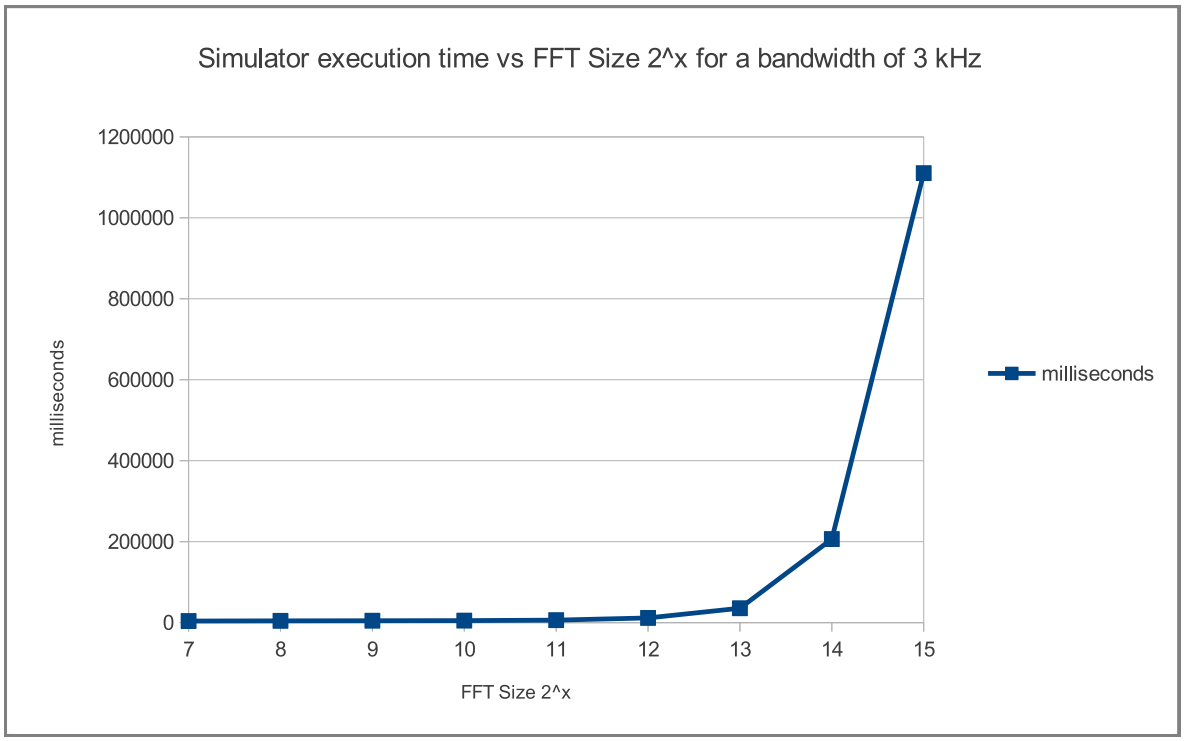


Figure 5.11: Execution time of the simulator vs FFT Size 2^x for a bandwidth of 3 kHz and simulation time of 10 Seconds.

the stage where the scheduling list is created.

The next experiment conducted was to determine the execution verses the processing bandwidths. It was observed that the simulator performed very well at lower bandwidths such as 3 kHz but as the bandwidths was increased, there was an exponential increase in the execution and the simulator fails at bandwidth greater than 3 MHz.

The final experiment which was conducted was to determine the performance of the simulator with increase in the number of pulsars. The simulator showed almost a linear increase in time as the number of transmitters were increased.

5.1.3.2 Hardware Requirement

In the previous section it was seen that the simulator didn't perform as expected for higher bandwidths. This is a classic case of computation requirements reaching out of bounds for a single processing system (in this case the laptop with specifications of Intel Core i73610QM CPU 2.30GHz 8 and 8 GB RAM). It may appear from the specifications of the system that it is a multi-core environment, but the SystemC-AMS doesn't exploit the multi-core functionality. On the other hand looking at the architecture of the receiver, it can be observed that most of the tasks can be implemented in parallel.

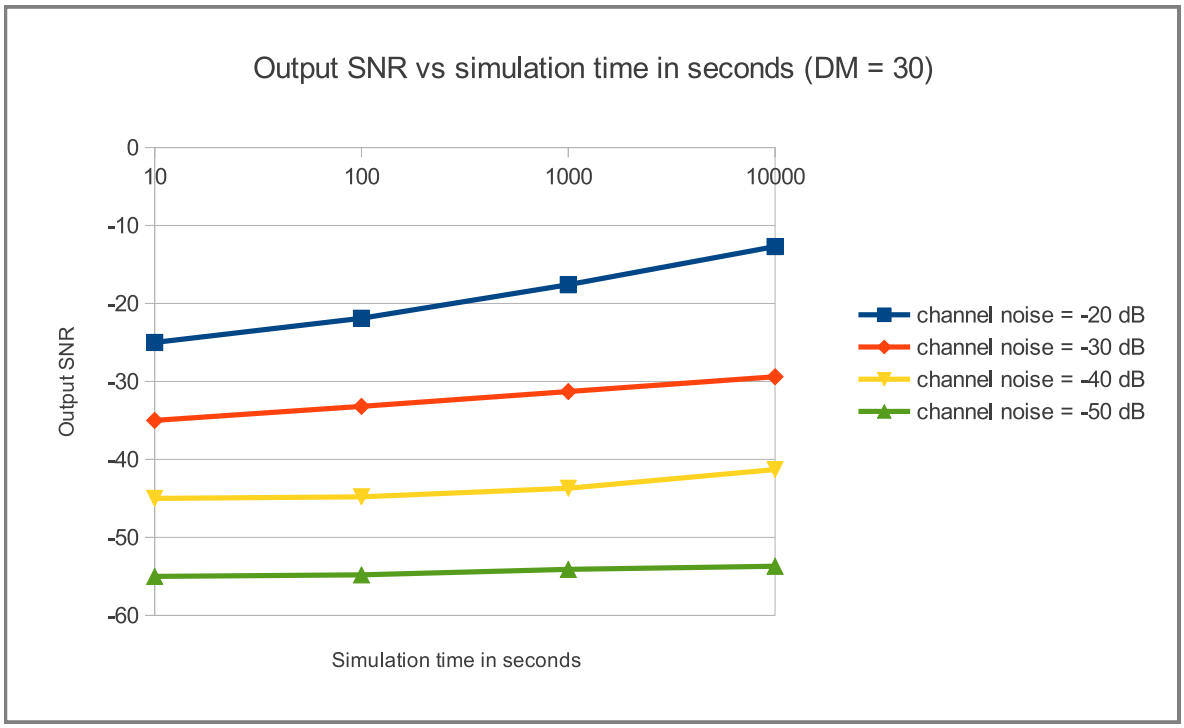


Figure 5.12: Output SNR vs simulation time in seconds.

Consider that the receiver receives signal at an input bandwidth of B_{in} and this bandwidth is divided into N frequency channels. At each output of these N frequency channels, a M point FFT and IFFT has to be performed. This translates to $4*N*M*log_2M$ cycles for processing. If the polyphase filter was not considered, the number of point FFT and IFFT required to perform would have scaled by $2*N*M*log_2M*N$ cycles. In terms of the exact number of FPGAs required and the memory bandwidth required, it is not possible to make an precise estimate of this quantity using the present simulator. Additional libraries need to created to extract the hardware requirement from the existing model. There would be a clear picture on these parameters once a synthesizable design for a polyphase filter and the architecture for performing FFT is finalized.

5.1.3.3 RaPSor

The second simulator was used to process the raw data from LOFAR. In Figure 5.17 the output of processing the raw data from one of the polyphase filter banks is shown. It was observed that the wide-band pulsar components appeared and more than one place which let to the conclusion that, this dataset was very noise with reflects visible. Never the less, the output of this simulation for folding periods of 200 s, 5000 s and 20000 s are shown in the following Figures 5.17a,5.17b and 5.17c. Upon further analysis of the dataset, it was observed that in the given dataset, the bandwidth of the dataset as mentioned in the information file is 0.1953125 MHz which should translate to a sampling frequency of $2*BW = 390625$ Hz. But the available sampling frequency of

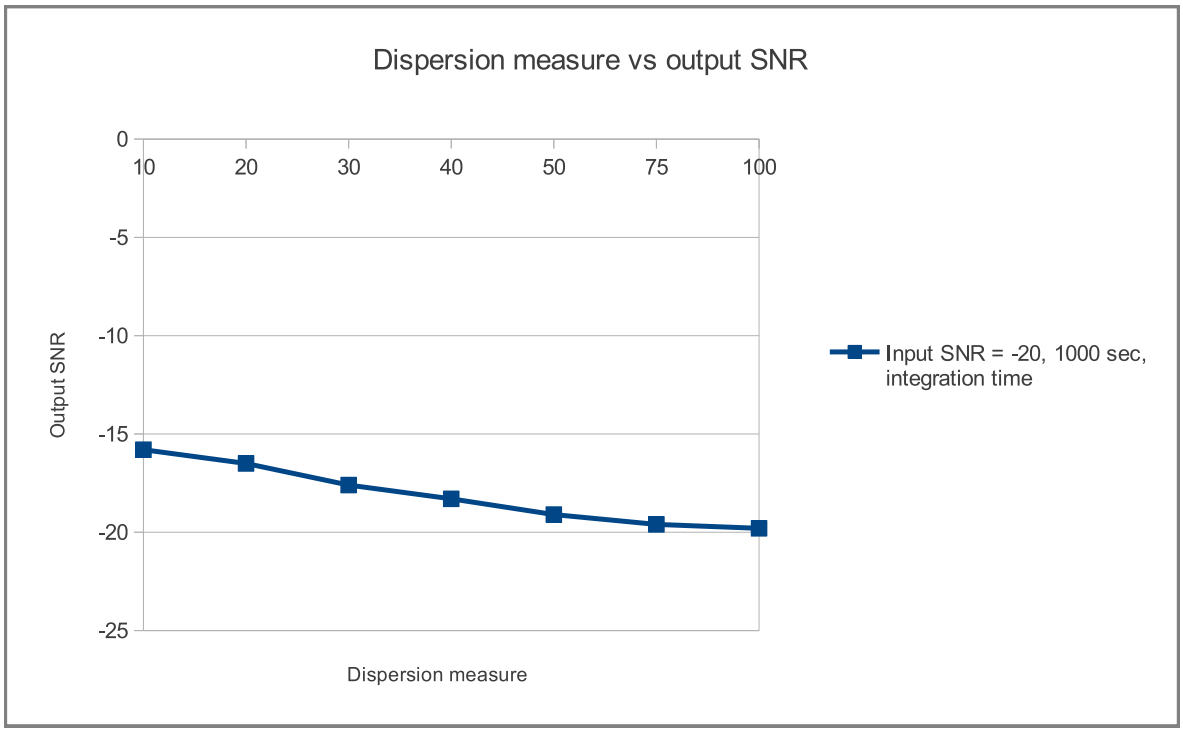


Figure 5.13: Output SNR vs Dispersion measure for an input SNR of -20 dB and simulation time of 1000 sec.

the dataset as mentioned in the information file is 762.9395 Hz which was downsampled by a factor of 512.

5.2 Limitations

In this section the limitations of the simulator are presented:

1. **High memory usage with increase in the FFT size:** It was observed that as the bandwidth of the generated pulsar signal increases, the size of the FFT in the channel model also has to be increased. As the size of the FFT used in channel model was increased, an exponential increase in the computation requirement was observed. As a result, the simulator failed to work at the desired frequency bandwidth of greater than 3 MHz. As a single FFT block, it was able to perform higher point FFTs, but as an integrated component within the simulator the total number of FFTs and IFFTs performed increased and as a result the simulator failed at the stage of scheduling. A possible solution to this problem could be to use standard scientific FFT libraries and writing a wrapper SystemC-AMS function around this library. The option was not explored due to the limited time frame.

2. **Automation:** The simulators were not completely automated as only the front-end was developed. The simulator can be made as a complete navigation system

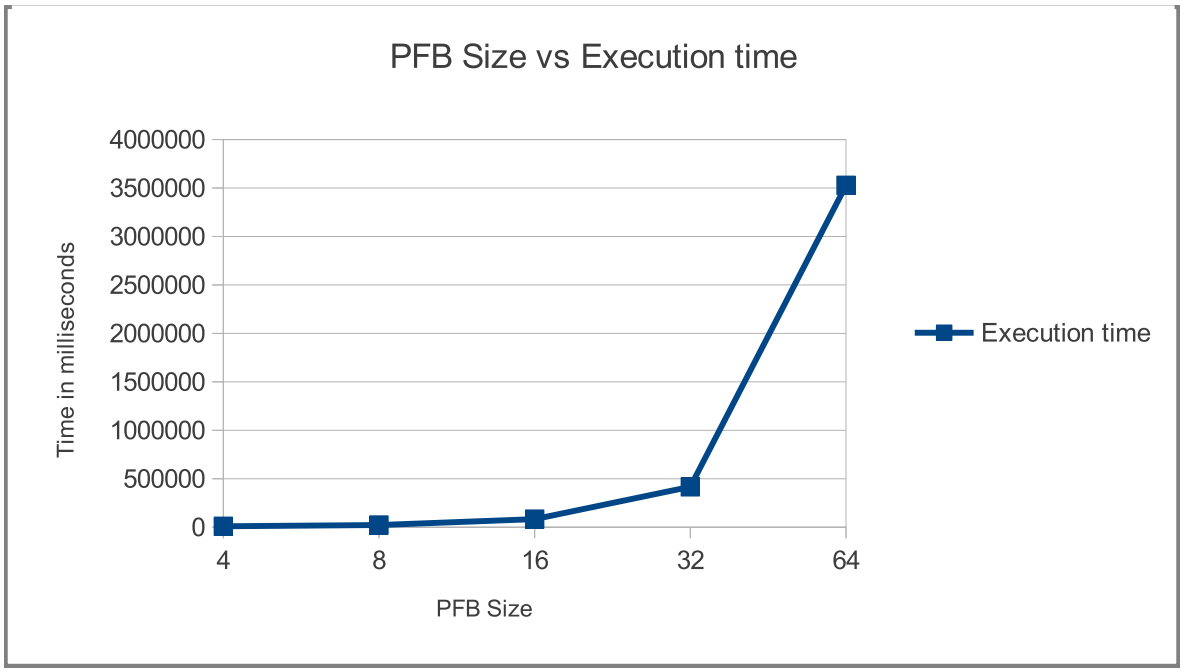


Figure 5.14: Size of poly-phase filter bank vs execution time.

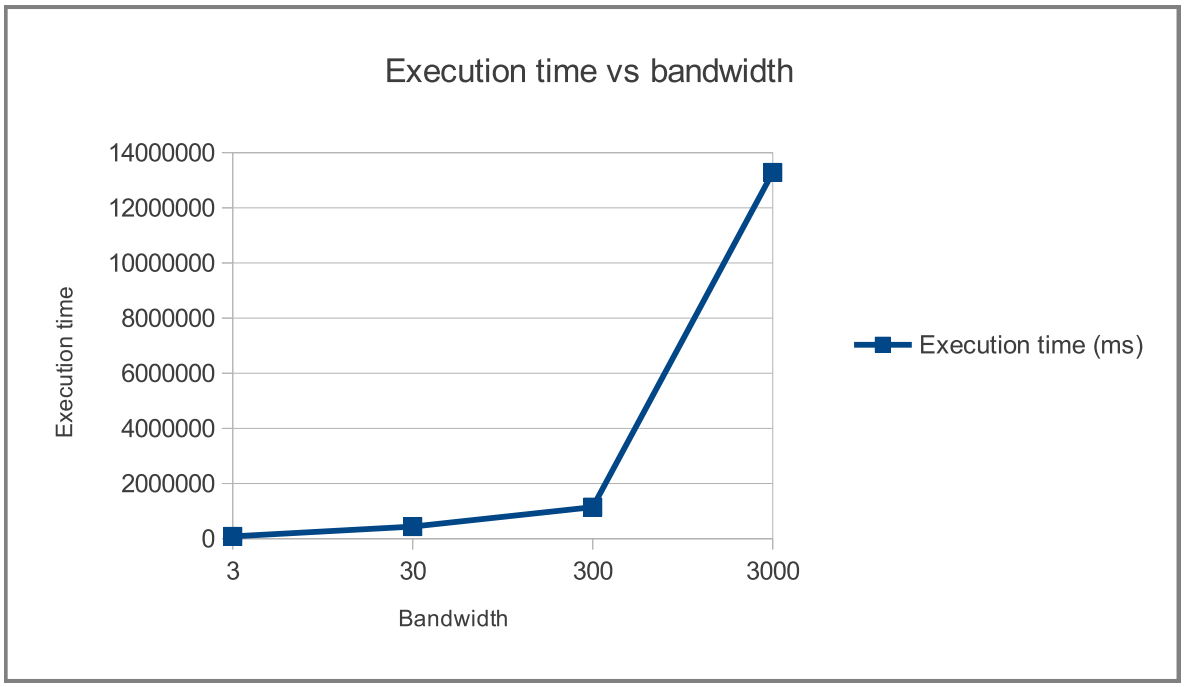


Figure 5.15: Execution time vs processing bandwidth.

by implementing the back-end system which performs the navigation algorithms.

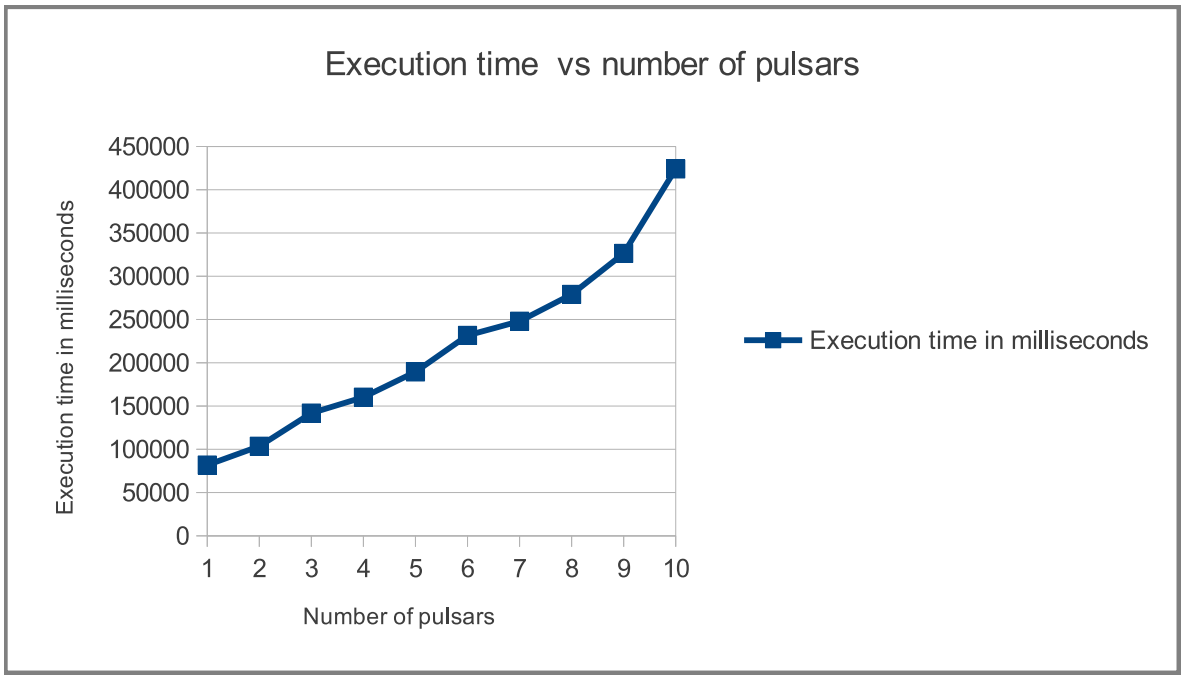


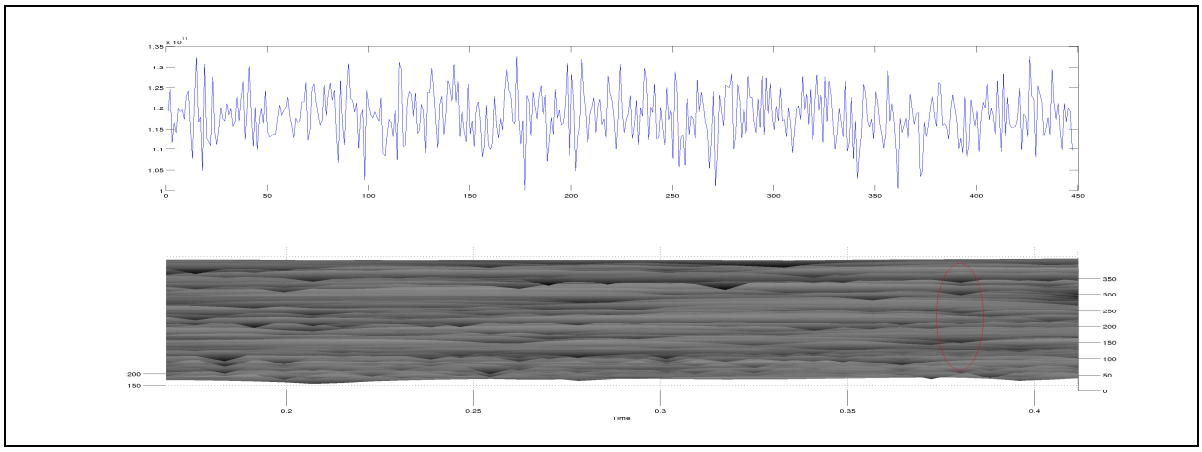
Figure 5.16: Execution time vs number of pulsar.

3. **Time to simulate:** The OSCI SystemC scheduler, schedules all the blocks in a sequential manner. Hence, for a more complex system such as SynRaPSor, the time to simulate is increases exponentially as the processing bandwidths are increased.

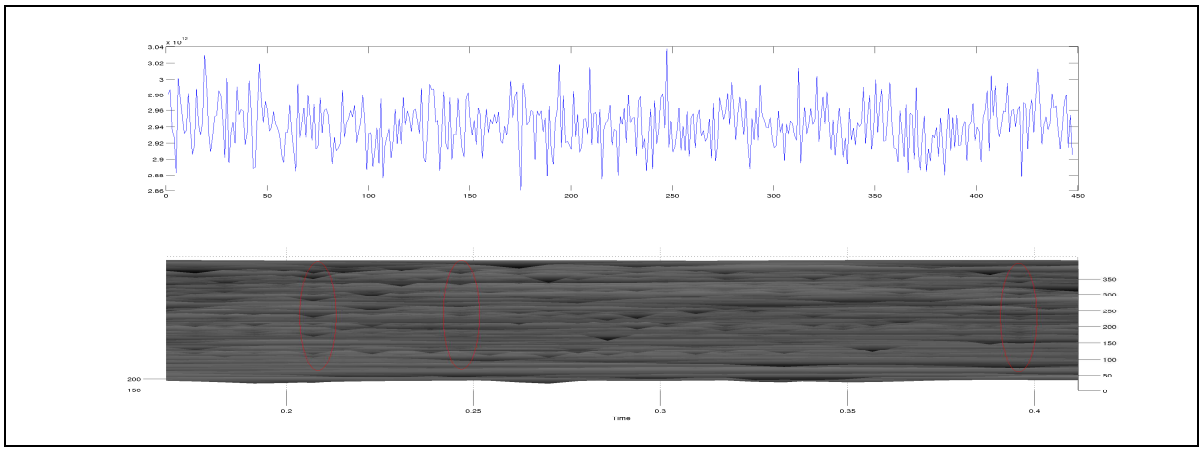
4. **Data Handling:** There aren't any available open source SystemC and SystemC-AMS IDEs, which would provide better data handling and representation features. So to interpret the outputs of the simulator, a parallel script has to be written to perform data acquisition and plotting. This slows down the performance of the simulator and also in a complex system with many signals, any new change or dynamically changing the scope of interest would not be possible.

5.3 Proposed Strategy and Road-map

Determining the Performance bounds: The design work carried out in this thesis was to provide a framework to test the design of a receiver which is very close to its actual hardware implementation. But during the implementation of this framework it was felt that some more investigation needs to be performed to determine the performance bounds of the system and provide a clear list of requirements to the front-end system. As these performance bounds are completely related to the algorithm followed to implement the navigation, it was not possible to determine the performance bounds using the front-end system.
Use of Matched filtering: From the present requirements list, it is evident that the

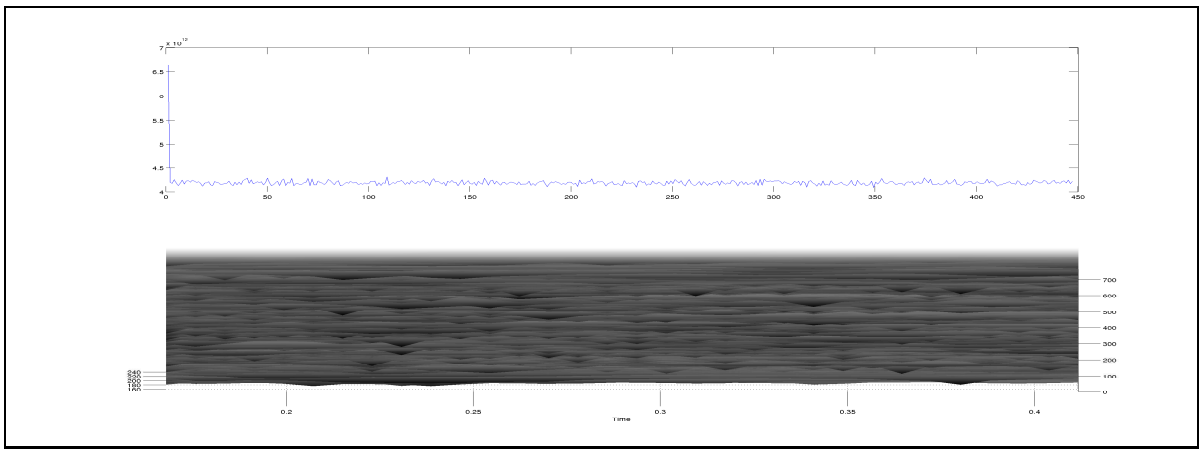


(a) Processing raw data through RaPSor after 200 seconds.



(b) Processing raw data through RaPSor after 5000 seconds.

Figure 5.17: Output of RaPSor which is plotted in Real-time.



(c) Processing raw data through RaPSor after 20000 seconds.

processing bandwidth that is considered is small enough to handle pulsar signals with lower dispersion measures without the need to implement a de-dispersion filter. But the de-dispersion filter has to be replaced with Matched filtering. Hence it is a very

interesting option to study the performance improvement in-terms of computational complexity and improvement in time for detection using Matched filtering (the origin of this idea was from discussions with Dr. ir. Gerard Janssen and from the paper [26]).
A scale down system: It was felt that a signal processing front-end capable of receiving all possible pulsars within the dispersion measures between 0 to 1000 was not required. As the dispersion measure increases, it was observed that integration time increased and from a logical way of looking at it, pulsars with higher dispersion measures are the signal which have traveled a longer distance. This indicates that the signal strengths could be very low as these signals are from pulsars which are farther away from us. So it is required to identify the band of dispersion measures to look for in a system meant for navigation. By restricting this band of observable pulsars a more realistic system in-terms of size and power can be achieved.
Synthesis the receiver blocks: The receiver blocks developed cannot be used for realtime processing. It can be used for offline processing of the received data. Hence it is required to develop synthesizable models which can be ported on real hardware.
From the above discussions and the results obtained so far, one of the critical observations that can be made is that, this system unlike the ones used for astronomical observations need not be able to detect all the available pulsars. Hence by restricting the system to observe only those pulsars which fall within the lower dispersion measure range, and improving the performance by using signal processing techniques, it is possible to reach the objective to developing a radio pulsar based navigation system.

Summary and Recommendations

6

In the first part of this chapter, a summary of the tasks accomplished in this thesis are provided. In the following section, recommendations for future work is provided.

6.1 Summary

The concept of a Pulsar based navigation system is a futuristic and revolutionary idea. This has the potential to replace GPS based navigation for terrestrial and space based navigation system (mainly in the LEO). But, it has many challenging hurdles to cross to be implemented in a scalable manner. This thesis was started with an idea to contribute to achieve a step closer to the final objective of developing an universal navigation system, that can make use of signals from pulsars.
When the thesis was started, the objective was to develop a hardware system based on the findings of [11] and [9] to process the raw data from the hardware setup in the campus of TU Delft. But as the development of this hardware setup to receive raw pulsar signals was also under development, at an early stage it was realized that the raw data from the actual hardware would not be available to test the hardware to be developed.
This prompted the idea to develop a hardware/software co-design based system, which performs the tasks of generating this pulsar data and processing it as well. To implement this system, a simulator was developed using open source C++ libraries, SystemC and SystemC-AMS. This tool provided the flexibility to describe the system similar to the hardware style (object oriented style) such as VHDL and combining it with the use of syntactically more powerful style of C++. Among the two libraries, SystemC-AMS was more suited to develop analog and mixed signal blocks and SystemC was suited to develop the digital blocks. As the development of a simulator using SystemC and SystemC-AMS to model signal from celestial objects was a first of its kind to be implemented, it required a thorough understanding of these astronomical objects and the stellar medium. From the survey, it was realized that there were many distinct properties exhibited by the pulsar and the stellar medium. To precisely take into account all these parameters in the implementation of the model would not be possible. So, a selected few properties and distortion effects that are dominant in the higher observing frequencies were considered for the implementation of the model.
A simulator (SynRaPSor) was developed analogous to a general communication system with the transmitter representing the pulsar, the channel effects induced by the inter stellar medium and the receiver representing the pulsar signal processing block. The system was considered as a baseband model to avoid the use of very high sampling rates. Each of the blocks described above were developed mathematically and then implemented using SystemC and SystemC-AMS. The simulator developed did not make

use of correlation blocks, so the performance was validated based on visual output of the signals. This required the use of pipes to send the output to other scripts in the background to display the data. This resulted in a very slow functioning of the simulator but never the less the outputs of the simulator was validated by analyzing the output. Some of the conclusions from this simulator is that, it can be used to generate a wide-band pulsar from more than one source and induce the effects of dispersion on these signals. The simulator was also checked from the point of view of the performance of the receiving block under the influence of different noise levels and some proposals were made to investigate these possibilities in the future. One of the limiting factor of the simulator was that it failed to function at higher bandwidths which limited us from checking the performance at the desired observation frequency of 1200 MHz and observation bandwidths of greater that 3 MHz.
A second simulator (RaPSor) was also developed to test the digital blocks of the receiver using the raw dataset available from the LOFAR. Experiments were conducted on this dataset for a period of 200 sec, 5000 sec and 20000 sec. It was concluded that there were sections of the spectrogram which suggests the presence of a wide-band signal, but it was not possible to extract the profile of the pulsar completely as it was observed that the wide signal appears at more than one place which is suggestive of noise to internal reflections in the antenna.
To conclude with, the outcome of this thesis has a large contributing factor in realization of the final system. The first simulator developed can be further modified to implement a few more influencing factors such as doppler effect (as the receiver is a moving object) and the back-end system which implements the navigation algorithms. These additions can be done easily as they can be added to the existing system as libraries. Then a simulator could be used perform real navigation and serve as a very important system engineering tool during mission design stage.

6.2 Recommendations and Future work

In the course of the design of these simulator several ideas were conceived that couldn‘t be realistically implemented given time frame. These are listed as the possible areas of future work:

1. **Process raw data from the actual setup:** Using the developed simulator, the raw data from the actual setup at TU Delft can be processed. But as the setup was not fully functional at the time of this thesis, it was not possible to perform this task. Hence task has to be performed as and when the hardware setup is able to receive signals.

2. **Feasibility of using Matched filtering:** Based on the functioning of a matched filter and the applications it is used for, it seemed to be a perfect candidate to be a part of the front-end system. But it was not possible to implement these models in SystemC-AMS to determine the increase in performance. Hence this is very interesting area of research for the future to check the improvement in performance of the system by the use of matched filtering.

3. **Implementing synthesizable models:** In the given time frame the synthesizable SystemC modules could not be developed. This can be carried out as a future work to implement the models developed in SystemC-AMS to synthesizable models which can be ported on real hardware.

4. **Radio Frequency Interference (RFI) Mitigation:** In the present models, RFI mitigation was not considered as it would not be needed in the space based system. But RFI mitigation algorithms need to be added to the existing framework to improve the performance of the earth based processing units (mainly for template creation).

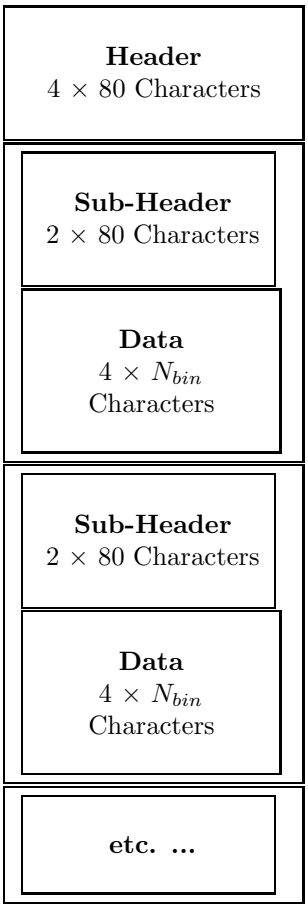
5. **Multi-beam forming phased array system:** The system considered in this thesis makes use of single parabolic dish to receive signals from the pulsars. It was felt that this method was inefficient for a navigation system. Hence the possible use of a phased array system with multi-beam forming capability would give the system a larger field of view and also mechanically a more stable system for a space application.

6. **GPU based implementation:** For the earth based observations which are required to create the pulsar template, it would be interesting to implement it using GPUs considering the performance gain and the flexibility it offers.

EPN data format

A

The following information was obtained from [27]



A.1 List of Allocated Variables

In the following tables a detailed description of all *header* and *sub-header* parameters is given.

(Header Status: March 25, 1996)

Name	Format	Comments
version		Logo + version of EPN format [EPN03.50]
counter		At the moment this is 4 (header) + 4× (2 (Sub-Header) + 1 (Data)) = 16 records (lines)
history		Space for comments, etc. (At the moment the filename [FN+filename] and the scan number [IC+scan number] are stored here)
jname	A12	Pulsar Jname (J2000.0), adopted from the Taylor, Manchester, Lyne catalogue of 706 pulsars
name	A12	Common pulsar name (B1950.0), adopted from the Taylor, Manchester, Lyne catalogue of 706 pulsars
P_{bar}	F16.12	Current barycentrc period, calculated from the EPOS header using the JHSPULS subroutine "prephead"
DM	F8.3	Pulsar dispersion measure, adopted from the Taylor, Manchester, Lyne catalogue of 706 pulsars
RM	F10.3	Pulsar rotation measure, adopted from the Taylor, Manchester, Lyne catalogue of 706 pulsars
CATREF	A6	Pulsar catalogue used during the observations, e.g [LMT95a]
BIBREF	A8	This is the bibliographical reference key for the publication of the data. It consists of the first letter of the authors' names + the year of publication, e.g. Seiradakis, Gil, Graham, Jessner et al. 1995, will appear as [SGGJ95] (UDP)
RA_{2000}	I2,I2,F6.3	R.A. of source, adopted from the Taylor, Manchester, Lyne catalogue of 706 pulsars
DEC_{2000}	I3,I2,F6.3	Dec. of source, adopted from the Taylor, Manchester, Lyne catalogue of 706 pulsars
telname	A8	Telescope- (or site-) name used for the observations (EFFELSBG)
EPOCH	F10.3	Modified Julian Date (MJD) of observation, adopted from the EPOS header (EPOCH = sngl(dayjul)
OPOS	F8.3	Position angle of the feed of the telescope. Not in header. Default value: [0.0] (UDP)
PAFLAG	A1	[A] = absolute position angle, else undefined
TIMFLAG	A1	[A] = absolute timestamps (UTC), else undefined
CDATE	I2,I2,I4	(*ddmmyyyy*) Date when the EPOS data were converted to EPN data. Adopted from the UNIX command "idate"
SCANNO	I4	Seq. number of the observation, adopted from the EPOS header "isn"
SUBSCAN	I4	Subscan Number, set to "1"
N_{pol}	I2	Number of polarisations observed. In the EPOS data this is usually the number of channels [4] (UDP)
N_{freq}	I4	Number of frequency bands per polarisation. In the EPOS data this is usually [1] (UDP).
N_{bin}	I4	Number of phasebins per frequency. In the EPOS data this is always [1024].
t_{bin}	F10.4	Duration (sampling interval) of a phase bin, adopted from the EPOS header "isamptm", taking into account the header change of 14/01/94
N_{int}	I6	Number of integrated pulses per block of data, adopted from the EPOS header "inttransm"
n_{cal}	I4	binno. for start of cal. signal. In the EPOS data this is always [1]
l_{cal}	I4	Length of calibration signal, calculated from the EPOS header"ical, isamptm", taking into account the header change of 14/01/94
F_{cal}	F8.3	Calibration factor for the data. [-1.0]. Please see *Notes* in page 4.

Name	Format	Comment
IDfield	A8	Type of data stream (I,Q,U,V etc.). Default value: [LHC, RHC, L-R, L-iR] (UDP)
n_{band}	I4	Ordinal number of current stream, [1, 2, 3, 4]
n_{avg}	I4	Number of streams averaged into the current one [1, 1, 1, 1]
f_0	F12.8	Effective centre sky frequency of this stream. Not in EPOS header. [1.41, 1.41, 1.41, 1.41] (UDP)
Δf	F12.6	Effective band width. Not in EPOS header. [40.0, 40.0, 40.0, 40.0] (UDP)
t_{start}	I12	Time of first phasebin with respect to EPOCH, calculated from EPOS header using the subroutine "preptime", t-start = (nstartime+3600) × 1000 + fractime × 1000
OFFSET	E12.6	Offset to be added to the data. Please see *Notes* in page 4.
SCALE	E12.6	Scale factor for the data. Please see *Notes* in page 4.
RMS	E12.6	RMS for this data stream. Please see *Notes* in page 4.
P_{app}	F16.12	Adopted from the EPOS header using subroutine "preptime", period1
$Data(1)$	I4	Scaled unsigned data for first bin
$Data(N_{bin})$	I4	Data for last bin of stream
IDfield	A8	Type of data stream (start of next sub-header)
...	...	...
...	...	...

Table A.1: Sub-Header and Data

B System Design

In this chapter, the system design of the receiver block using SystemC is provided. This chapter is not included as a part of the main thesis as the complete design was not completed during the course of the thesis. But, an insight has been provided towards a final realization of the complete receiver system.

B.1 SystemC Design of the Digital blocks of the Receiver

In the previous chapters of this thesis, the architecture of the signal processing front-end of a receiver unit has been described using SystemC-AMS in Chapter 4. The same top level system design of the receiver can be used to further develop the SystemC architecture of the design. A SystemC design of the receiver is more closer to a realizable final hardware.
In the SystemC design, addition care needs to be take with signaling and use of clocks. A top level overview of the system architecture of the receiver is provided in the Figure B.1. It can be seen that, as individual components, the most complex operation is performed by FFT/IFFT. This leads to the conclusion that, the performance of the system is directly proportional to the performance of the FFT/IFFT block. Some of the other individual components required to realize this receiver are, lowpass FIR filtering, SDRAM and a memory controller.

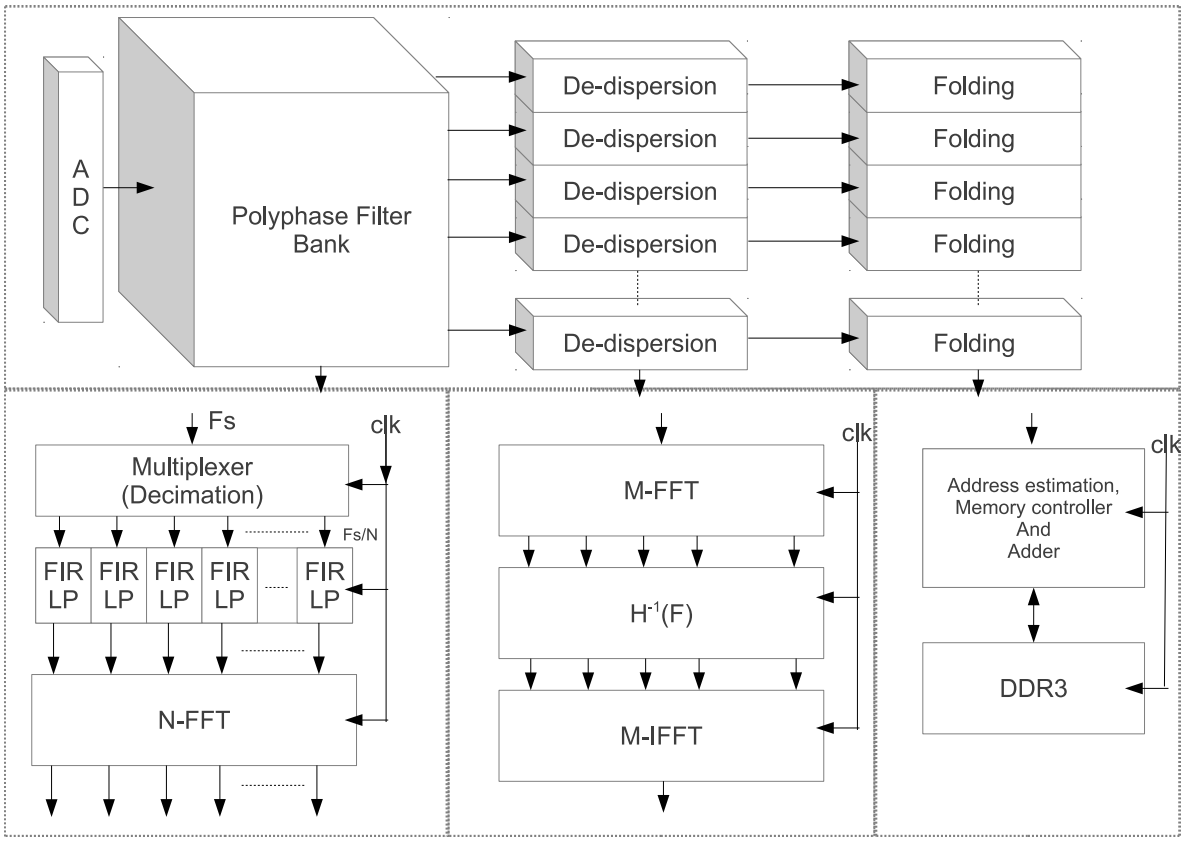


Figure B.1: A top level view of the receiver architecture in SystemC

B.1.1 Poly-phase filtering

The input signal is channelized using a polyphase filter bank. The filter bank is used to divide the input bandwidth into smaller frequency bands. The details of the SystemC implementation of this filter given in Figure B.2. It was observed that this design was not synthesizable due to the FFT library used. Hence, it is required to perform modifications to the existing design to realize a synthesizable Polyphase filter.

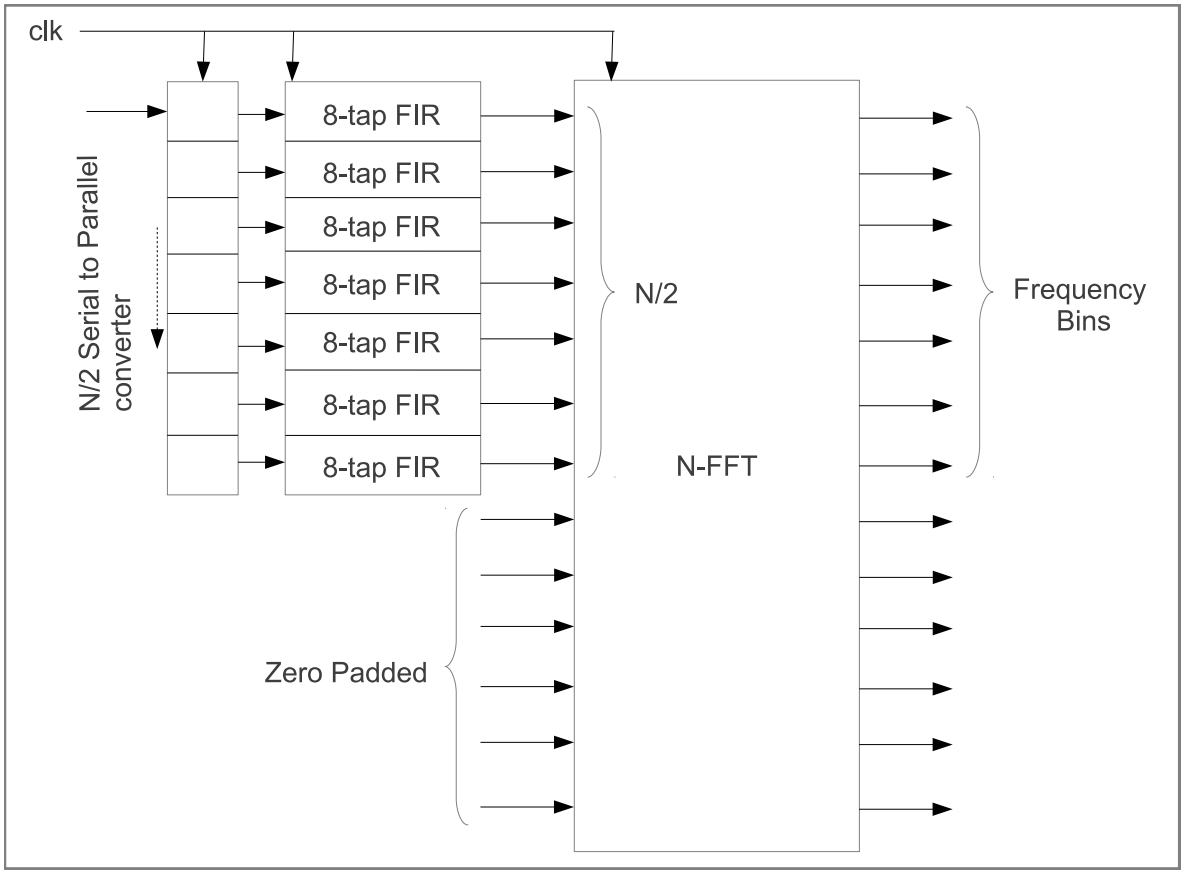


Figure B.2: SystemC Architecture of the Polyphase filter bank

B.1.2 De-dispersion

The De-dispersion process considered in this architecture consists of frequency domain multiplication of the filter coefficients with the input signal. This requires the use of FFT and IFFT blocks to convert the incoming signal into its frequency domain and back to the time domain after phase correction. Considering the use of a standard FFT/IFFT library, the systemC implementation is straightforward. Considering a dataset of complex input samples of size N, a 2N point FFT is taken to convert these samples to the respective frequency domain. The frequency domain output is now multiplied with the complex filter coefficients $H^{-1}(f)$ of size 2N with the last N coefficients padded with zeros. This phase corrected frequency domain signal is converted back to its time domain signal using an IFFT.

B.1.3 Folding

Folding is performed using SystemC by initializing an external memory (XRAM) and the input samples from each of the PFB output is alloted a bank (a set of standard locations). A size of 4096 KiloBytes was considered for each frequency bin and the input signal is converted to its phase and stored in the respective phase bin. This can be considered as the address representing 0^{th} byte represents a phase of 0 degrees and the data in

$4,194,304^{th}$ Byte represents a phase of 360 degrees. The folding module also needs to consider for offsets which are required to be considered as in the case of Barycentric corrections. This offset is passed as parameter θ to the folding block (computed by the back-end) and the corresponding offset address is calculated.
In the final system, the folding block can be considered as a simple DDR memory controller which reads the data from the address calculated by the folding algorithm and adds the incoming data to this data and writes back to the same location. Details about the Barycentric correction is given in the following section.

B.1.3.1 Barycentric correction

Ideally the receiver is in an inertial frame assuming that the pulsar is also in an inertial frame. But the receiver when inside earth can be considered as rotating with the angular velocity of the earth. The earth in turn is orbiting around the sun. Hence the received signal needs to be corrected for the errors caused due to this motion.

B.2 Summary

The receiver system which needs to be realized in real hardware was designed using SystemC based on the SystemC-AMS models described in the previous chapters. It was observed that the design was not easily synthesizable (contrary to what was expected) using the CtoS tool. Hence, it was not possible to complete the synthesizable design during the given time frame and can be considered for the future work.

Bibliography

[1] M. Kramer, "Looking Inside a Neutron Star, lighthouse beam of the radio pulsar." http://www.jodrellbank.manchester.ac.uk/news/2000/neutronstar/, December 2008.

[2] M. Kramer and D. R. Lorimer, "Handbook of Pulsar Astronomy." http://www.jb.man.ac.uk/research/pulsar/handbook/figures.html, 2004.

[3] J. Sala, A. Urrela, X. Villares, and R. Estalella, "Feasibility Study for a Spacecraft Navigation System relying on Pulsar Timing Information," tech. rep., ESA, June 2004.

[4] I. H. Stairs, E. M. Splaver, S. E. Thorsett, D. J. Nice, and J. H. Taylor, "A Baseband Recorder for Radio Pulsar Observations," Mar. 2000.

[5] A.A.Kestilä, S. Engelen, E. Gill, C. Verhoeven, M. Bentum, and Z. Irahhauten, "An Extensive and Autonomous Deep Space Navigation System using Radio Pulsars," in *61st International Astronautical Congress, Prague, CZ.*, 2010.

[6] N. R. Manchester, B. G. Hobbs, A. Teoh, and M. Hobbs, "ANTF Pulsar Catalogue ." http://www.atnf.csiro.au/research/pulsar/psrcat/, 1993-2006 (2005).

[7] J. R. Wertz, "Autonomous Navigation and Autonomous Orbit Control in Planetary Orbits as a Means of Reducing Operations Cost," July 2003.

[8] *GPS versus LORAN-C for Vehicular Navigation in Urban and Mountainous Areas*, 1993.

[9] S.Engelen, "Deep Space Navigation System Using Radio Pulsars Front-End," Master's thesis, Delft University of Technology, Aug. 2009.

[10] A. Hewish, S. Bell, J. Pilkington, P. Scott, and R. Collins, "Observation of a Rapidly Pulsating Radio Source," *Nature*, vol. 217, pp. 709–713, 1968.

[11] V. K. Chaudhri, "Fundamentals, Specifications, Architecture and Hardware Towards a Navigation System Based on Radio Pulsars," Master's thesis, Sept. 2011.

[12] M.Kramer and D.R.Lorimer, *Handbook of Pulsar Astronomy.* CAMBRIDGE UNIVERSITY PRESS, 2005.

[13] D. Lorimer, "Radio Pulsar Statistics," in *Neutron Stars and Pulsars* (W. Becker, ed.), vol. 357 of *Astrophysics and Space Science Library*, pp. 1–17, Springer Berlin Heidelberg, 2009.

[14] P. A. G. SCHEUER, "Amplitude Variations in Pulsed Radio Sources," *Nature*, no. 218, 1968.

[15] *Multifrequency observations of radio pulse broadening and constraints on interstellar electron density microstructure* , no. 605, 2004.

[16] T. H.Hankins and B. J. Rickett, "Pulsar Signal Processing," vol. 14, p. 55, Academic Press, 1975.

[17] R. Karuppusamy, B. Stappers, and W. van Straten, "PuMa-II Recent Observations,"

[18] R. Karuppusamy, B. Stappers, and W. van Straten, "PuMaII: A wide band pulsar machine for the WSRT," Feb. 2008.

[19] "Collaboration for Astronomy Signal Processing and Electronics Research ." Februrary 2012.

[20] R. DuPlain, S. Ransom, P. Demorest, P. Brandt, J. Ford, and A. L. Shelton, "Launching GUPPI: the Green Bank Ultimate Pulsar Processing Instrument,"

[21] Kestila and A. Alexander, "Deep Space Navigation System Using Radio Pulsars Back-End," Master's thesis, Delft University of Technology, Aug. 2009.

[22] *An Introduction to Modeling Embedded Analog/Mixed-Signal Systems using SystemC AMS Extensions*, June 2008.

[23] "SystemC-AMS TUV AMS Library." http://www.systemc-ams.org/BB_library.html.

[24] M. Barnasconi and C. Grimm, "SystemC AMS extensions Users's Guide," tech. rep., NXP Semiconductors, TU Vienna, 2009.

[25] J. Ou, P. Brunmayr, F. Muhammad, J. Haase, and C. Grimm, "TU Vienna SystemC AMS Communications Library," tech. rep.

[26] R. Heusdens, S. Engelen, P. J. Buist, A. Noroozi, P. Sundaramoorthy, C. Verhoeven, M. J. Bentum, and E. Gill, "Match filtering approach for signal acquisition in radio-pulsar navigation," in *63rd International Astronautical Congress (IAC 2012), Naples, Italy*, (Paris. France), p. B2.6.10, International Astronautical Federation (IAF), October 2012.

[27] A. G. Lyne, A. Jessner, and N. D'Amico, "The Structure of EPN Datasets V3.5," tech. rep., Nuffield Radioastronomy Laboratory, Jodrell Bank; Max Plank Insititute of Radioastronomy; Istituto di Radioastronomica del CNR, March 1996.