DESIGN AND IMPLEMENTATION OF AN IRNSS RECEIVER ACQUISITION BLOCK

A Project Report

submitted by

KARTHIK JAYACHANDRAN

in partial fulfilment of requirements
for the award of the dual degree of

BACHELOR OF TECHNOLOGY AND MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS

May 2018

THESIS CERTIFICATE

This is to certify that the thesis titled **DESIGN AND IMPLEMENTATION OF AN IRNSS RECEIVER ACQUISITION BLOCK**, submitted by **Karthik Jayachandran**, to the Indian Institute of Technology, Madras, for the award of the degrees of **Bachelor of Technology and Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. V. Kamakoti
Research Guide
Professor
Dept. of Computer Science
IIT-Madras, 600 036

Place: Chennai

Date: 10th May 2018

ACKNOWLEDGEMENTS

This project would not have been possible without the support and guidance of a lot of people. Firstly, I would like to thank my guide Prof. V Kamakoti, for taking me under his wing and providing the opportunity to work at the RISE Lab in order to learn and explore the field of hardware design. This experience has proved both educational and fun. I would also like to thank Mr. Arjun Menon, for his constant support, guidance and valuable inputs, without which completing this project would not have been possible. I also wish to extend my gratitude to my project mates Pranavkumar Shende and Yashodhara Tarey, for their ever-present help and ideas. A special shout-out goes to my friends, for making these five years an amazing, life-changing experience that I'll never forget. Lastly, a big thank you to my family, for always being there for support and motivation.

ABSTRACT

KEYWORDS: GPS; IRNSS; Acquisition; Tracking; CORDIC; Correlator; PRN.

Navigation receivers form a pivotal of modern technology, with their widespread use in a multitude of products and industries. This project aims to design and construct a hardware synthesizable design for the Acquisition block of a modern-day navigation receiver, which can interchangeably be used for both the GPS and IRNSS services. The Acquisition block forms the initial stage of data processing for this, providing an initial estimate of the satellite(s) acquired based on the incoming signal, which can further be smoothed using tracking loops. This system is built to interface with a separated in-phase and quadrature input stream based RF front end module.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

BSV	Bluespec System Verilog
IRNSS	Indian Road Navigation Satellite System
DSP	Digital Signal Processing
PRN	Pseudo Random Noise
IF	Intermediate Frequency
RF	Radio Frequency
BRAM	Block Random Access Memory
CORDIC	Coordinate Rotation Digital Computer

NOTATION

y_i	In Phase Signal Component
y_q	Quadrature Signal Component
f_d	Doppler Frequency
f_c	Carrier Frequency
C_{rep}	Local Code Replica
N_s	Samples per Code
n_d	Number of Doppler Frequency bins

CHAPTER 1

INTRODUCTION

1.1 Receiver Overview

A navigation receiver is designed to acquire and demodulate data transmitted from satellite(s) in order to find its own location. The receiver as a whole is composed of 3 major components, as shown by the block diagram in Figure 1.1. The RF front end receives signals from all visible satellites, and is responsible for down-converting it to an appropriate Intermediate Frequency (IF). Also, it samples and thereby digitizes the received analog signals. The digitized IF signals are then passed to the Digital Signal Processing (DSP) block of the receiver where data is demodulated and pseudo-range measurements, i.e. Code Phase and Doppler, are carried out. The measurements and demodulated data are then passed on to the Navigation Processor where the final estimation is carried out.

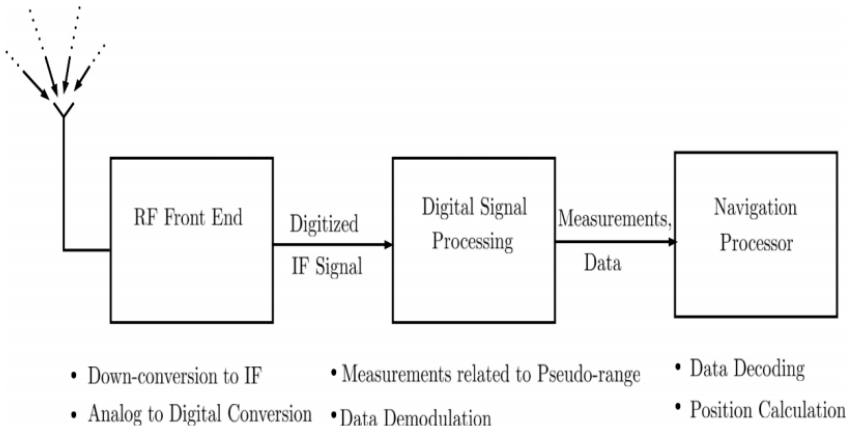

Figure 1.1: Navigation Receiver block diagram.

This project designs and implements the Acquisition module, which is a part of the Digital Signal Processing (DSP) block of the Navigation Receiver, using Bluespec System Verilog (BSV). The overall function of this block, as described above, is to make measurements on the received signals and demodulate it for estimation of exact navigation data.

1.2 DSP Overview

The Navigation Receiver's Digital Signal Processor has two main processing blocks - Acquisition and Tracking. After initialization of the receiver and its data, the Acquisition block determines which of the available satellites is acquired by the receiver. Signal acquisition can be done in one of two ways, based on availability of previous measurement estimates (either no information is known, or an approximate Doppler is known). The data is then processed by the tracking block, which is responsible for converting the coarse estimates provided by the acquisition block into finer estimates, and keeping track of incoming code phase, carrier phase and Doppler frequency.

This design allows the end user to configure the signal properties (sampling frequency, intermediate frequency, data type, file location etc.) and system properties (selection of file reading parameters, acquisition and tracking parameter selection). The primary input can be from a simulated or recorded location file, or an actual antenna.It has been configured to use data which has separated in-phase and quadrature components, for example, a data stream provided by an AD9361 RF antenna front-end setup.

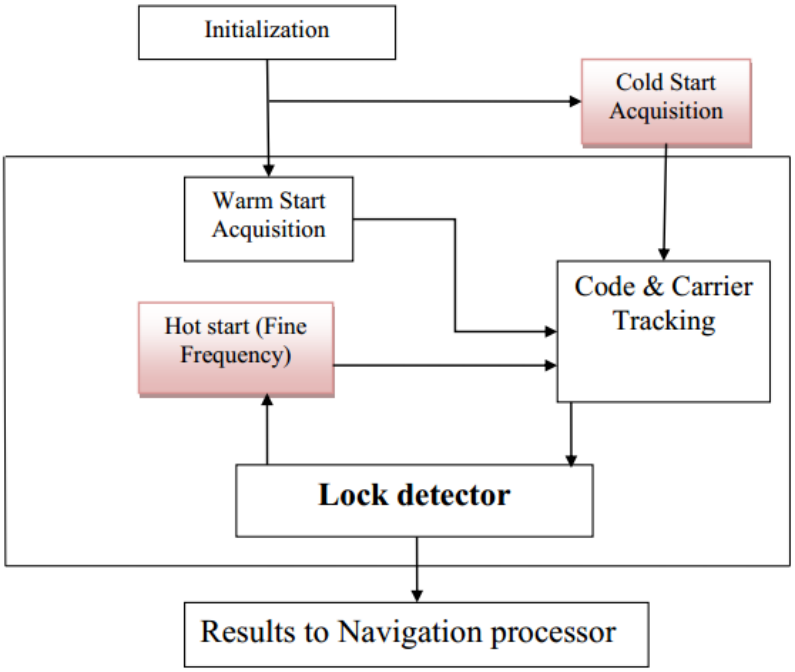

Figure 1.2: Code flow of the Baseband Processor

1.3 Bluespec System Verilog

BSV(BSV, 2003) is a high level Hardware Description Language that defines synthesizable behaviour using rules - an assertion expressing a potential atomic state transition. The BSV compiler produces efficient RTL code that manages all potential interactions between rules by inserting appropriate arbitration and scheduling logic rather than having to manually handle scheduling. Modules can be linked using two tools: interfaces and methods, and the provision of flexible and useful library elements like FIFO's, BRAM's etc. are an important feature of this language. It has powerful static type checking which removes potential human errors, enabling elimination of errors in the compiling stage itself. The versatility in BSV's type parametrization, which allows for modules and functions to parametrize each other, leads to easier design re-usability and flexibility in linking modules. The BSV compiler also can generate the synthesizable Verilog code(s) from developed Bluespec code(s) which can be used later directly for synthesis purposes.

CHAPTER 2

Theory and Mathematics of Acquisition

2.1 Input Description

The Acquisition block of the navigation processor is responsible for an initial estimate of the signal code phase and Doppler frequency via the process of correlation. The inputs consist of the digital signals obtained using the RF front end, down-converted to an IF. They are essentially a pair of signal streams for in-phase and quadrature data, both of which are the multiplication of the navigation data bits, PRN code chips and the respective frequency carrier. This is in line with the Standard Positioning Service (SPS) satellite signal structure. The Navigation data bits have a rate of 50 samples per second(sps) . PRN code chip rate is defined as 1.023 Mcps, giving us a time length of 1ms (code length is 1023 chips). IRNSS signals have 2 carrier frequencies, L5, i.e. 1176.45 MHz from the L-band and 2492.08 MHz from the S-band.

2.2 Theoretical Description

The process of acquisition entails correlation in order to calculate the code phase of the signal, which involves sample-wise multiplication of two signals and accumulation of the products. Here, the PRN used is a Gold Code. The first step involves performing a base carrier wipe-off, depending on whether the RF front-end does it or not. Then, a two-dimensional search is carried out to find the correct code phase and Doppler for the given PRN code by preparing a correlation value grid (Ward, 1996).

The 2-D correlation grid is computed by first expanding the PRN code to a larger size in order to match the incoming data sampling rate. The incoming signal is then correlated with a smaller frequency in order to account for the Doppler shift (chosen as a range of ± 10 KHz, and medium range step size) around the IF. This range of frequencies is taken as one axis for the grid, with the other axis being all possible values

of code shift. At each combination of frequency and code shift, a single summation value is computed by finding the correlation of the in-phase and quadrature data streams separately, squaring and then adding them. The two highest peak values, with a certain code phase range excluded in the second peak search to account for noise, are then found and a ratio of the two is computed in order to determine whether the given satellite is acquired or not. The peak computation procedure is then repeated with a smaller frequency step size and the same code phase for fine-tuned Doppler results.

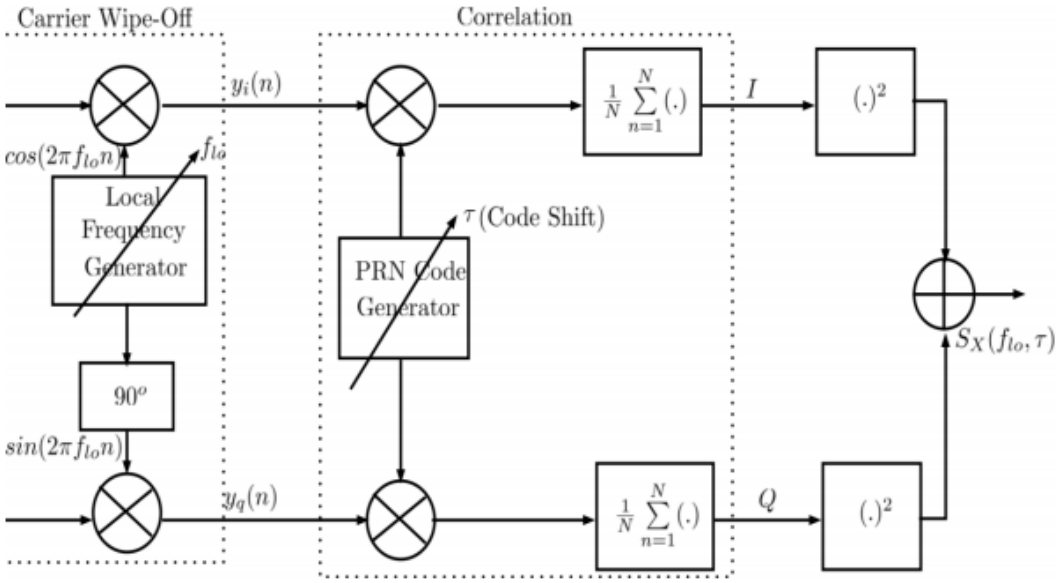

Figure 2.1: Acquisition Correlator block diagram.

2.3 Mathematical Description

The IRNSS transmitted signal stream can be described with 3 components for the PRN, data and carrier being multiplied, defined as follows:

$$d(t) = \sum_{i=-\infty}^{\infty} d(i) rect[t - \frac{T_d}{2}] \tag{2.1}$$

$$p(t) = \sum_{i=-\infty}^{\infty} p(i) rect[t - \frac{T_c}{2}] \tag{2.2}$$

$$c(t) = \cos(2\pi f_c t) \tag{2.3}$$

where T_d is the data bit time of 20ms, T_c is the chip time period of $1/1.023Mcps = 0.9775\mu s$, and f_c is the carrier frequency. This gives us the expression for the combined transmitted signal as $x(t) = d(t)p(t)c(t)$. At the receiver end, the signal received is of the form

$$y(t) = \sqrt{P} d'_{T_d}(t - t_0) p'_{T_{chip}}(t - t_0) \cos(2\pi f'_c(t - t_0)) \tag{2.4}$$

where t_0 is the travel time from satellite to receiver, $f'_c = f_c + f_d$, where f_d is the Doppler frequency. The signal is then down converted to the IF, which is represented as

$$y_d(t) = y(t) \cos(2\pi(f_c - f_{IF})t) \tag{2.5}$$

On substitution of $f'_c = f_c + f_d$ and $\theta_0 = -2\pi f'_c t_0$ and simplifying, we get

$$y_d(t) = \sqrt{P} \frac{d'_{T_d}(t - t_0) p'_{T_{chip}}(t - t_0)}{2} [\cos(2\pi(2f_c + f_d + f_{IF})t + \theta_0) + \cos(2\pi(2f_d + f_{IF})t + \theta_0)] \tag{2.6}$$

Ignoring the high frequency components due to filtering by the RF front end, we now have

$$y_d(t) = \sqrt{P} \frac{d'_{T_d}(t - t_0) p'_{T_{chip}}(t - t_0)}{2} \cos(2\pi(2f_d + f_{IF})t + \theta_0)] \tag{2.7}$$

The last pre-processing step is to digitize this signal using the front end's ADC, which yields

$$y_d[n] = \sqrt{P} \frac{d'_{T_d}[nT_s - t_0] p'_{T_{chip}}[nT_s - t_0]}{2} \cos(2\pi(2f_d + f_{IF})nT_s + \theta_0)] \tag{2.8}$$

This design also accounts for the in-phase and quadrature components of the signal coming in separately from the front end, which then results in

$$y_i[n] = y_d[n]\cos(2\pi f_{lo} n) \tag{2.9}$$

$$y_q[n] = y_d[n]\sin(2\pi f_{lo} n) \tag{2.10}$$

In order to perform the acquisition, a rotated replica code is required, which is represented as $C_{rep}(n-\tau)$, where τ represents the code shift, running from 1 to the length of the expanded PRN code. This then gives us our correlation results as

$$I = \frac{1}{N}\sum_{n=1}^{N} y_i[n]C_{rep}(n-\tau) \tag{2.11}$$

$$Q = \frac{1}{N}\sum_{n=1}^{N} y_q[n]C_{rep}(n-\tau) \tag{2.12}$$

Storing them as a single peak value, we get $S_\chi(f_{lo}, \tau) = I^2 + Q^2$. We repeat this procedure in order to fill the acquisition grid, and then find the highest peak. After this, the frequency is fixed, and a second highest peak is found outside of a excluded range. If the determined Acquisition ratio is above the set threshold, then the satellite is considered to have been acquired, the results stored and passed onto the tracking module.

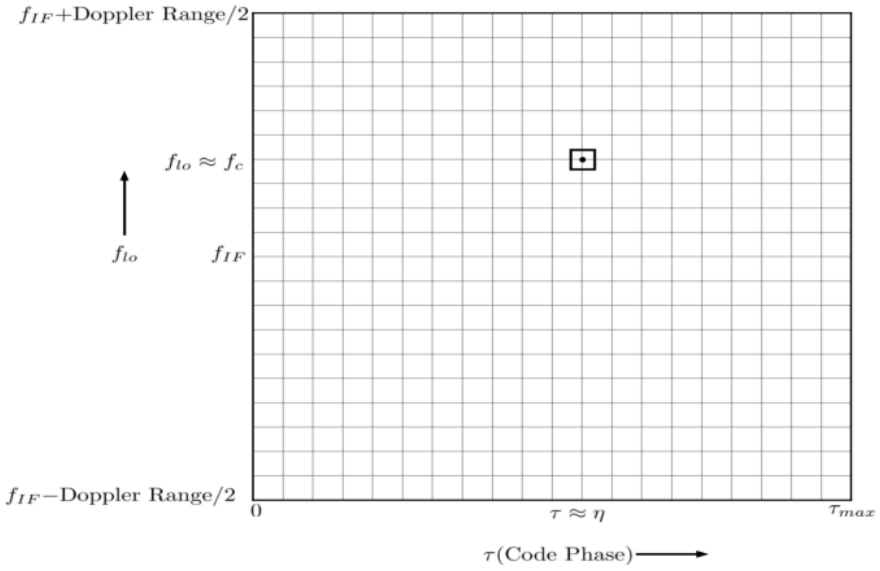

Figure 2.2: Acquisition grid

CHAPTER 3

Algorithm and Implementation in BSV

3.1 Challenges faced due to hardware

The acquisition algorithm explained in Chapter 2 requires the computation of $N_s \times n_d$ correlations, where N_s is the number of samples per code of the expanded PRN and n_d is the number of steps chosen for computing the Doppler shift in the range of ± 10 KHz, plus a finer frequency search of n'_d within the bigger frequency step to estimate a better Doppler estimate. Within this, each of these correlations involves the multiplication and accumulation of $2N_s$ pairs of data (since the operation needs to be performed for both the in-phase and quadrature components of the signal). This is then followed by the computation of the two highest peaks, which involves a linear search depending on the sizes of data and number of frequency steps involved. As a result, a traditional loop or loop-like structure, while reasonable for simulation purposes using a traditional programming language, viz. Matlab or Python, will end up utilizing too much in terms of hardware resources, as this approach will create $N_s \times n_d \times 2$ adders and multipliers.

3.2 Algorithm in BSV

3.2.1 Doppler shift correlation

The first step of processing, assuming that the RF front end takes care of carrier wipe-off, is to perform a correlation with the current Doppler frequency bin value. Although this can be achieved by a normal correlator by providing the input data streams and a generated pair of sinusoidal signals as inputs, the quicker and more efficient way to do this is to use a CORDIC(Coordinate Rotation Digital Computer) block.

3.2.2 Introduction to CORDIC

CORDIC, or Volder's Algorithm (Volder, 1959), is a mathematical algorithm that describes an efficiently implementable method to compute trigonometric functions and other mathematical operations purely using additions, subtractions, table lookups and shift operations. It runs iteratively on a basic set of equations obtained from the rotation matrix:

$$x_{i+1} = x_i cos\theta - y_i sin\theta \tag{3.1}$$

$$y_{i+1} = x_i sin\theta + y_i cos\theta \tag{3.2}$$

On taking a common factor of $cos\theta$, choosing $\theta = \tan^{-1}(2^{-i})$, where i is the iteration count, we obtain the following final equations:

$$x_{i+1} = x_i - y_i d_i 2^{-i} \tag{3.3}$$

$$y_{i+1} = y_i + x_i d_i 2^{-i} \tag{3.4}$$

$$z_{i+1} = z_i - d_i tan^{-1}(2^{-i}) \tag{3.5}$$

where $d_i = \pm 1$, and i indicates the iteration count. Here, the sign of z_i determines d_i, which indicates the direction of rotation with respect to the previous x_i and y_i values to compute the $(i+1)^{th}$ iteration value. Based on the initial values chosen for x,y and z, various functions can be realized.

For example, in order to compute $sin\theta$ and $cos\theta$, the choice of initial values is $x_0 = 0.607253$, $y_0 = 0$, $z_0 = \theta$. The initial value of x_0 is chosen as so due to the appearance of a product of cosines of $\tan^{-1}(2^{-i})$ in the expression above. Choosing the iteration count to be 16 for a reasonable degree of accuracy, we get $K = \prod_{i=0}^{15} \cos(\tan^{-1}(2^{-i}))$. After the computation, we get $y = sin\theta$ and $x = cos\theta$.

For the case of the Doppler correlation, we require $y_i \cos\theta$ and $y_q \sin\theta$, so we run the CORDIC with a different initial value set, basically $x_0 = Ky_i$ or Ky_q, with $y_0 = 0$ and $z_0 = \theta$. The x and y components respectively give us the cosine and sine products required.

3.2.3 Correlation algorithm

The solution to the problems stated in section 3.1 is to parallelize and parametrize the correlation procedure; the former allows for re-usability of hardware, and the latter makes the implementation flexible based on the nature of the input and availability of hardware. Parallelization of the entire acquisition block is carried out by parallel correlations, simultaneously computing peaks and handling correlation on different sets of data, and .

First, the basic correlation is performed in elementary **Correlator modules**, which consists of 4 multipliers and a 4-input adder. This module is then repeated a certain number of times, based on the availability of hardware and user preference. This set of correlator instances is known as the **Correlator bank**(Kong, 2017), which essentially performs $n \times 4$ multiplications and returns n sums simultaneously, where n is the number of correlator modules in this bank. These individual sums are then accumulated in order to obtain the result of correlation of a $4n$ - sized section of the total code phase - signal correlation.

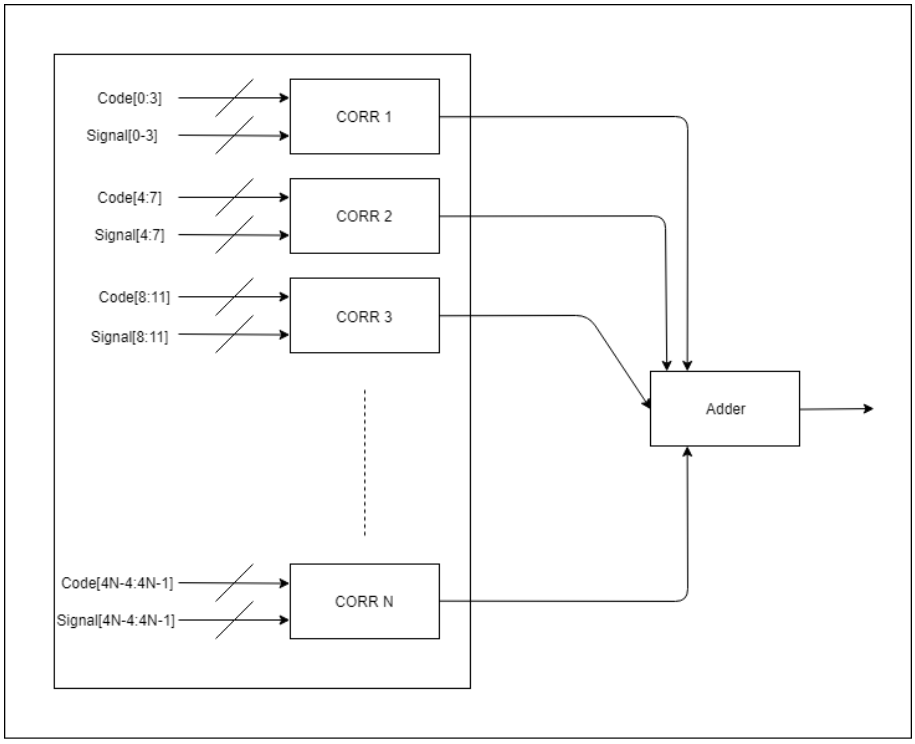

Figure 3.1: Correlator Bank

This Correlator bank is then reused a number of times, depending on the actual size of the data and bank, and the final correlated sum of a particular code shift is obtained. This entire procedure is simultaneously carried out for both the in-phase and quadrature streams to obtain 2 sums. Finally, the in-phase and quadrature correlation sums are squared and added in order to obtain a single value that is stored (indicated by a single point in Figure 2.2).

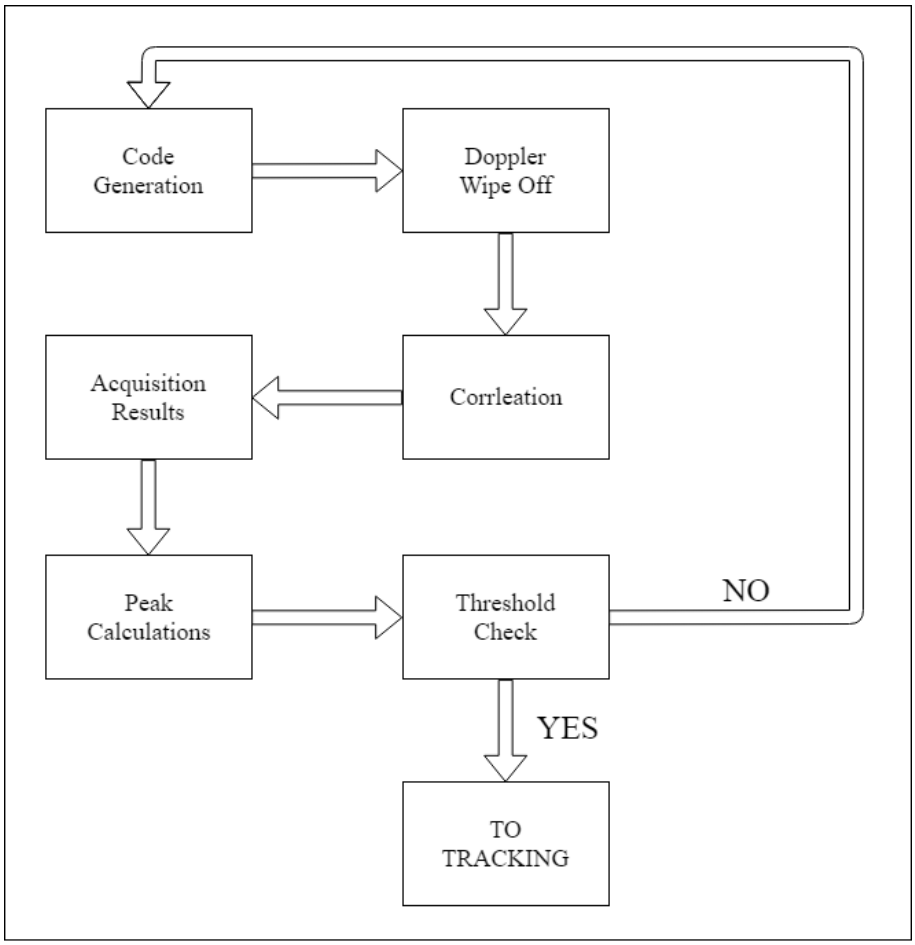

Figure 3.2: Overall Acquisition Code Flow

The next step involves finding the two maximum peaks in each code shift. The previous correlation steps will be run for an arbitrary number of code shifts (say, n_c) before the peak finding process can be initiated. The peak finding section initiated by parsing the current set of n_c correlation sums for a maximum value, which is stored along with its index. The next step involves establishing the exclude range before parsing the current set again for the second highest value. This range is set to 1 code chip wide either side of the maximum value's location. This is done to account for noise effects around the peak causing possible false peaks to show up.

The process described for n_c sums is then repeated for all code shifts at a given frequency, while maintaining a global maximum and second maximum for all values so far. The global maximum peak, i.e. the one across all Doppler frequency bins, is then procured, and the second maximum for that particular frequency is taken as the second maximum, not the global second.

The final step to check if a satellite is acquired is to take the ratio of the two peaks computed so far, for the given satellite PRN code, and check if it is above the Acquisition ratio threshold, a parameter that needs to be set carefully in order to correctly filter out the wrong satellites and indicate the correctly found ones. If this step is successful, the frequency bin is further fine-tuned by fixing the code phase and varying the Doppler frequency with a much smaller step size around the initial estimate, and re-calculating the peaks for this particular range of frequencies and code phase. This process gives us a much more precise estimate of the Doppler frequency. The acquired results are stored and sent forth for the tracking algorithm.

CHAPTER 4

DESCRIPTION OF CODE BASE

Here is a brief description for each of the modules in the acquisition block:

correlator : The correlator enumerates the basic module. It takes 4 values of the current code phase and 4 signal values as input, and returns the multiplied and accumulated value as an output. ($sum = c_1y_1 + c_2y_2 + c_3y_3 + c_4y_4$)

correlator_Bank : It takes 3 Vectors of size $4N$ as input, one for the code phase, and 2 sets of signals (in-phase and quadrature), and uses interfacing to create multiple correlators. Simultaneously computes correlation in 2 banks for both signals and returns the partial accumulation as the result.

acquire_Bank : The above correlator banks need to be run so as to cover the entire code base. This module has access to the entire data stream, and sequentially sends parts of data to the 2 correlator banks.

code_Generator : The initial PRN code available is of a fixed size, i.e. 1023 bits of data. Based on the sampling rate, this module expands and copies the PRN data into a larger set of bits to allow for correlation to take place.

cordic_Rotation : CORDIC allows us to iteratively compute $y_i \cos\theta$ and $y_q \sin\theta$ in an efficient way. The inputs here are values of $y_i[n]$ or $y_q[n]$ and the requisite value of $2\pi f_d t[n]$. Output obtained is $y_i[n] \cos\theta_n$ or $y_q[n] \sin\theta_n$.

write_Data : This is the first module in the entire code flow, which takes in each bit of signal data as the input and stores it into memory for Acquisition to take place.

peak_Finder : After finding a certain number of in-phase and quadrature sums, this module is fired to start computing the first 2 peaks. It is a structurally reusable module that enables simultaneous computation of some peaks while correlation is taking place.

prn_Finder : On getting a certain index indicating the PRN number, this module obtains and sends back the correct PRN gold code.

acquisition_Control : The central module responsible for controlling overall code flow by performing the code phase rotations and creating and computing Doppler frequency steps for both coarse and fine searches.

The following is the simulated acquisition grid, coded and plotted using MATLAB:

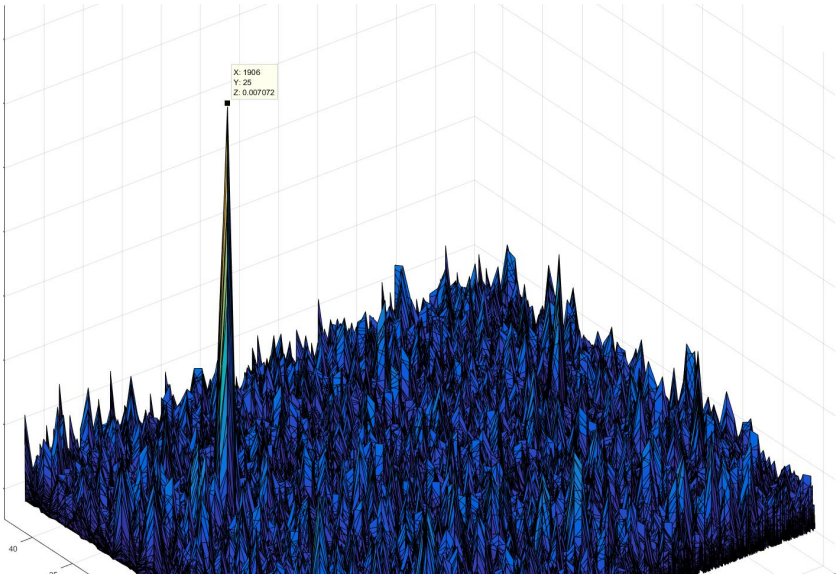

Figure 4.1: 3-D plot of acquisition grid

CHAPTER 5

CONCULSION

5.1 Summary

This project has built a parallelized and parametrized acquisition module for a navigation processor. The optimizations implemented via parallelization of the computation and code flow both enables for otherwise too large data to be processed and operated upon with relative ease, and the use of efficient concepts such as CORDIC results in a less complex hardware realization of the algorithm. The acquisition algorithm implemented here is also efficient in its usage of storage, memory and hardware, reusing correlator banks and storing only the original code shift by initializing memory reads from different points rather than performing a manual rotation.

5.2 Future Work

This implementation can be used as a packaged module, and coupled with a developed tracking module, can be used as a standalone navigation DSP module, to be integrated with a processor and antenna for real-time navigation data information.

REFERENCES

1. **Kong, S.-H.** (2017). High sensitivity and fast acquisition signal processing techniques for gnss receivers: From fundamentals to state-of-the-art gnss acquisition technologies. *IEEE Signal Processing Magazine*, **34**(5), 59–71.

2. *BSV, 2003* (retrieved:). URL `http://www.bluespec.com/`.

3. **Volder, J. E.** (1959). The cordic trigonometric computing technique. *IRE Transactions on electronic computers*, (3), 330–334.

4. **Ward, P. W.**, Gps receiver search techniques. *In Position Location and Navigation Symposium, 1996., IEEE 1996*. IEEE, 1996.