FPGA Implementation of the Front-End of a DOCSIS 3.0 Receiver

A Thesis Submitted to the
College of Graduate and Postdoctoral Studies
in Partial Fulfillment of the Requirements
for the degree of Master of Science
in the Department of Electrical and Computer Engineering
University of Saskatchewan
Saskatoon

By

Rory Gowen

Head of the Department of

Electrical and Computer Engineering

University of Saskatchewan

3B78 Engineering Building

57 Campus Drive

Saskatoon, Saskatchewan S7N 5A9

Canada

OR

Dean

College of Graduate and Postdoctoral Studies

University of Saskatchewan

116 Thorvaldson Building

110 Science Place

Saskatoon, Saskatchewan S7N 5C9

Canada

Abstract

The introduction of cable television (CATV) in the 1940s and 1950s has significantly influenced communications technology. Originally supplying only one-way television programming, the CATV industry recognized the potential of two-way communications. Starting with the introduction of pay-per view services in the 1980s, two-way communications over CATV networks eventually expanded into supplying internet access services. The increased demand for CATV services, and thus the increased demand for CATV equipment, has led the CATV industry to develop interoperability standards. The primary standard now used by the CATV industry is the Data Over Cable Service Specification (DOCSIS). DOCSIS defines both the upstream (data towards the CATV provider) and downstream (data towards the CATV customer) transmission channels. This includes specifications for the modulators and demodulators used in these channels.

The number of manufacturers of CATV modulators and demodulators has greatly increased over the last twenty years and continues to do so. As the number of competitive CATV equipment suppliers increases, these manufacturers must look to ways to remain competitive by reducing time-to-market and costs associated with equipment design, as well as allowing their designs to be flexible so that they may adapt to the improvements in DOCSIS.

In the past, manufacturers have primarily used Application Specific Integrated Circuits (ASICs) to implement digital hardware designs for CATV equipment. ASICs have a very high initial setup cost and do not allow for system modifications without a complete redesign. Recently, Field Programmable Gate Array (FPGA) technology has been introduced that allows manufacturers to both modify their designed digital hardware structures without a complete physical hardware redesign, as well as providing a reduced initial setup cost. Although in the long term, ASICs provide a cheaper alternative to FPGAs when produced in quantity, FPGAs provide quicker time-to-market in new product development and allow changes to made after initial release. This ability to change designs after release and the quicker time-to-market has led manufacturers to adopt FPGAs in new products.

A critical component in the upstream channel of a DOCSIS compliant system is the Quadrature Amplitude Modulated (QAM) receiver. The data received at the QAM receiver have undergone several impairments including additive noise, timing offset, and frequency and phase mismatches between the transmitted modulated signal and the signal received at the demodulator. It is the function of the front-end of the receiver to correct for these impairments.

This thesis presents methods for, and an example of, the design and implementation of a DOCSIS compliant QAM receiver front-end that corrects for timing, phase and frequency impairments experienced in the upstream communication channel when additive noise is present. The circuits presented are designed and implemented to reduce hardware costs when using FPGA technology. In addition, the circuits designed do not use proprietary logic, which gives designers more flexibility when implementing their own demodulator front-end circuitry.

The FPGA implementation presented in this thesis achieves an average MER of 54.3 dB in a no-noise channel and close to 31 dB MER in a 25 dBc AWGN channel. The overall design uses 65 dedicated 18-bit by 18-bit multipliers and 2,970 bytes of RAM to implement the digital front-end of the receiver.

Acknowledgements

I would like to thank my supervisor Dr. Brian Daku for helping me achieve the goal of completing this thesis and helping me throughout the writing process. I would also like to thank Dr. Eric Salt for his insight into research areas of interest to industry and for helping me to choose topics to include in this thesis.

I would also like to thank my parents for their love and encouragement, without which I would not be the person I am today. You have both allowed me to follow my dreams and have given me the opportunities needed to do so.

Finally, to my wife, Tamzen, thank you for your love and support that has helped me during the writing process and continues to do so throughout all my endeavors.

Contents

List of Tables

List of Figures

List of Abbreviations

ADC	Analog to Digital Converter
ASIC	Application Specific Integrated Circuit
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
CATV	Cable Television
CIC	Cascaded Integrator-Comb filter
CM	Cable Modem
CMTS	Cable Modem Termination System
CORDIC	COordinate Rotation DIgital Computer
DAC	Digital to Analog Converter
dB	Decibels
dBc	Decibels with respect to carrier power
DC	Direct Current
DOCSIS	Data Over Cable System Interface Specification
DSP	Digital Signal Processing
FDMA	Frequency Division Multiple Access
FIR	Finite Impulse Response
FPGA	Field Programmable Gate Array
HDTV	High-definition Television
IEEE	Institute of Electrical and Electronics Engineers
IIR	Infinite Impulse Response
ISI	Inter Symbol Interference
LFSR	Linear Feedback Shift Register
LO	Local Oscillator
LPF	Low Pass Filter
LUT	Look-Up Table
MAC	Media Access Control
MER	Modulation Error Ratio
NCO	Numerically Controlled Oscillator
NLPAP	Near Linear Phase All-Pass filter
PHY	Physical Layer
PLL	Phase Locked Loop
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying
RAM	Random Access Memory
RF	Radio Frequency
ROM	Read only Memory
SRRC	Square Root Raised Cosine
TDMA	Time Division Multiple Access
TV	Television

Chapter 1
Introduction

1.1 Cable Television Background History

In the 1940s and 1950s television made its debut to the general public. Television broadcasts were sent over the air from local transmission stations to TV sets owned by end users. Unfortunately the over-air transmissions did not have a great enough range to service all the users that demanded TV services. This excess demand gave rise to the cable television (CATV) industry.

Initially, entrepreneurs devised methods by which large antennas could be placed at elevations above cites and towns where demand for TV services was high. These antennas could receive the TV broadcast signals that could not normally be picked up in lower lying areas. Well shielded coaxial cables were then connected to the antennas and fed into the the lower lying demand areas. Initially the cable drops were only run to a select few locations such as TV stores and hotels. However, the demand continued to increase and soon the CATV service was being run to private individuals as well. These locations were charged a one time fee, that gave them ownership rights over the cable that was run to their TV, as well as a monthly maintenance fee. The fees that were collected were then used to build additional CATV antennas, amplification hardware and other required infrastructure, as well as to maintain the existing infrastructure [29].

Although initially developed for use in areas with poor over-the-air signal reception, the demand for CATV systems in densely populated areas with decent signal reception increased. Over-air reception of signals in large cities with high-rises and many larger buildings caused signal reflection and reception degradation of TV broadcast signals. The CATV service was an answer to reception problems in these densely populated areas.

The use of CATV in the highly populated areas caused a new problem. Some areas of the cities had very strong signal reception from over-the-air broadcast stations. If these areas had CATV drops run to their TVs, the strong over-the-air signal could be picked up by the TV set even though there was no antenna connected to that set. The set would then display the over-air broadcast as well as the signal on the cable connected to the TV. This may not seem like a problem, however, the coaxial cable caused slight delays in the signal being received at the TV due to the length of the cable run. This delay would cause two out of sync broadcasts to be displayed on the TV at the same time. CATV companies were forced to develop a solution for this double signal reception issue. One such solution was to create a TV converter box that would take existing TV broadcasts and position them at different frequencies. The converter box would then place the desired channel (signal) at a frequency that was not locally used and the TV set would be permanently set to receive at this frequency. It was then up to the user to tune to the desired broadcast using the converter box as opposed to their TV set, giving rise to equipment similar to that which is used today.

By the start of the 1970s there were more than 4.5 million CATV subscribers [29], with such a high increase in numbers of subscribers over a relatively short period of time and even higher numbers predicted in the future, CATV and other companies began to realize the potential of two-way communications over the CATV network. Throughout the 1970s, companies such as Vicom and Sears as well as the education industry began developing two-way communications systems that could be used for conferencing, ordering products and distance education [16].

During the 1980s most one-way cable services were being replaced with two-way systems. These two-systems allowed the cable providers to supply higher quality services to customers, such as pay-per-view TV. Although these systems seem primitive by today's standards they encouraged the advances towards today's two-way CATV services.

Ever since the public release of the world wide web, the demand for access to this information medium has been steadily increasing [31]. The cable service providers viewed this demand as an opportunity to gain additional subscribers by providing internet access over cable. Cable service providers upgraded their existing coaxial cable distribution system creating hybrid coaxial and fiber optic cable distribution systems that allowed for higher data bandwidths, and reduced maintenance costs, resulting in the CATV networks used today [24, 18].

1.2 CATV Network topology

Modern CATV networks must provide the ability to both send and receive data in order to supply on demand services. A modern CATV network has a fairly simple topology. The CATV provider, referred to as the headend, transmits data downstream to the customer and the customer can in turn transmit upstream data to the headend. The CATV network consists of a cable modem termination server (CMTS) in the headend which feeds the customer's cable modem (CM) via a coaxial or hybrid coaxial/fiber link as shown in figure 1.1.

Two-way communication over a single coaxial cable network necessitates the division of the cable frequency spectra used for the upstream and downstream communications. Due to the large amount of data that must be sent to customer's CM units for services such as high-definition TV (HDTV), internet, and pay-per-view TV, the downstream side of the CATV network is given the largest bandwidth range (100 MHz up to 1 GHz). The upstream is given a far smaller frequency bandwidth that can be between 5 MHz and 85 MHz [7].

The bandwidth of the upstream is further divided into a set of channels so that multiple CM units can transmit data to the CMTS at different frequencies and data rates. The bandwidth of each of these channels is based upon the selected data transmission rate that has been negotiated between the CMTS and CM units during the initial setup of a CM unit.

As the CATV network structure has evolved to meet the demands for bi-directional communications and primarily for internet access, the upstream portion of the CATV network has undergone the largest number of changes to allow for users to send and request larger packets of information at higher rates.

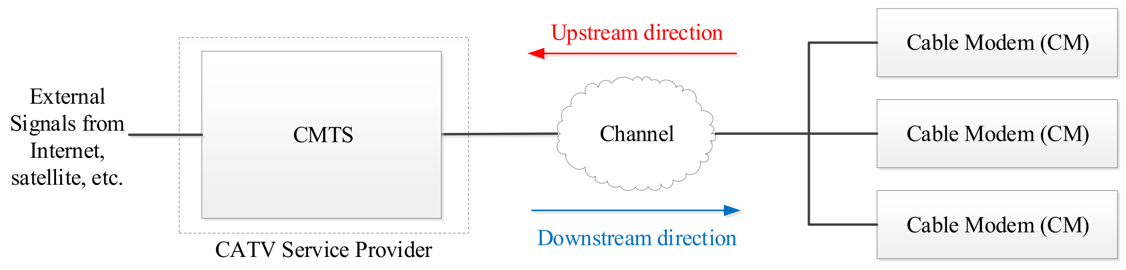


Figure 1.1: CATV Network High Level Overview

1.3 DOCSIS

During the infancy of the CATV networks there were no network standards or regulations. The lack of standards and regulations, since cable companies were in different geographical areas, led to each company developing their own amplifiers, converter boxes and other related signal transmission and reception equipment. As the demand for internet access grew, the need for cable providers to be able to interface with one another became apparent, requiring some sort of standardization between providers.

By 1998 two standards were being developed: the IEEE 802.14 and the Data Over Cable System Interface Specification (DOCSIS) standards. The DOCSIS standard was created from partnerships between several North American CATV providers [10]. The reason for this partnership was to provide a faster standardization process and quicker time-to-market turnaround for CATV products than was available with the IEEE 802.14 [18].

The DOCSIS standard is maintained by CableLabs, a non-profit organization setup by the cable companies. The organization was originally set up to ensure the compatibility of cable equipment between various cable service providers and has now expanded to providing testing and certification of cable equipment.

Since its inception in the late 1990s, the number of CATV service providers using DOCSIS to implement and verify interoperability between devices made by different manufacturers has greatly increased. According to Fellows and Jones, in 2001 CableLabs had certified over 100 modem models from 36 different companies [10]. As of this writing, CableLabs has certified over 1000 modem models from well over 100 different manufacturers and now has over 50 member companies from the United States, Canada, Mexico, South America, Europe, Asia and Australia [1, 2].

DOCSIS defines protocols that are required to interface between the cable modems (CM) and cable modem termination servers (CMTS). The protocol stack includes the physical layer (PHY) and the media access control (MAC) specifications, among others. The scope of these protocol layers is well beyond that of this thesis, however some brief explanation of the importance of the mentioned protocols will be given to help aid in the understanding of their need.

1.3.1 Physical Layer Protocol

The physical layer of the DOCSIS protocol stack sets forth the communication modulation schemes that may be used over a cable network as well as worst-case channel specifications that may be encountered during data transmission and reception. This specification also outlines the frequency ranges for both the upstream and downstream transmissions.

1.3.2 Media Access Control Protocol

In a CATV network there may be many CM units and each of these CMs may need to transmit data back to the CMTS to request services to be provided. DOCSIS specifies that these transmissions may use either frequency division multiple access (FDMA) or time division multiple access (TDMA) channels.

The MAC layer of the protocol is used to control transmissions in the upstream (from CM units to the CMTS) direction. It is the job of the MAC in the CMTS to allocate transmission frequencies and times to individual CM units for their use when transmitting data upstream.

For TDMA channels, the CMTS MAC must allocate specific times for each CM to transmit data and for how long each CM may transmit. The MAC in the CM must keep track of these transmission times and frequencies in order to send data back to the CMTS at the appropriate moment.

In general, the MAC layer is responsible for keeping track of transmission times, timing synchronization, transmission power levels, transmit frequencies and dealing with transmission collisions (two or more CMs transmitting on the same frequency at the same time) between units in the cable network.

1.4 FPGAs in CATV Systems

The evolving standards for the CATV industry and high demand for new and faster technology from customers have caused CATV providers to find methods of producing equipment that can easily be modified to keep up with changing standards.

Large manufacturers have tended to use application specific integrated circuits (ASICs) to produce customer equipment in large volumes. ASICs tend to be very power efficient and cost effective when producing large quantities of the ASIC designs. The downside of ASIC based products is that the ASIC, once designed and manufactured, cannot easily be changed to accommodate a new standard without full redesign. In addition, ASCICs tend to have a very high initial cost of manufacturing.

In recent years, CATV and telecommunication companies have turned from ASIC to field-programmable gate array (FPGA) for their modem implementations. FPGAs tend to have higher power requirements than that of ASICs [20] and somewhat higher large volume costs than an ASIC based design. However, FPGAs have a much lower initial setup cost and can be reconfigured to add additional features to accommodate changes in the evolving CATV standards without the need to completely redesign an ASIC.

Costs associated with FPGAs tend to be based on the number of logic elements and multipliers in the FPGA. As a general rule when producing FPGA based designs the multipliers tend to be in high demand. The greater the number of multipliers on an FPGA the higher the cost on the FPGA since the multipliers also tend to cost much more per unit than the cost of other logic elements.

In order to reduce the hardware costs associated with producing CATV equipment and to keep up with the evolving standards to remain competitive in the market, manufacturers of CATV equipment must look for ways to drive down production costs. This thesis will look at methods to implement a DOCSIS based upstream demodulator front-end with a focus on techniques that can reduce the number of resources required for FPGA implementation. The thesis will also examine the performance of these implementation schemes.

1.5 Research Objective

The primary objective of this research is to develop procedures, which designers and rese-archers can use, to aid in the selection and implementation of filters and recovery circuits used in the digital front-end of a CATV quadrature amplitude modulated (QAM) receiver suitable for implementation on a FPGA device.

The performance of a FPGA implementation, using the methods presented, is carried out and analyzed, according to the DOCSIS 3.0 specifications, in order to provide a guided design example demonstrating the use of the these methods. The performance of the implemented demodulator must meet the CATV standards supplied in DOCSIS 3.0 given in terms of modulation error ratio (MER).

The procedures and implementation methods presented here are done with emphasis on reducing required FPGA resources. The resources available on the selected Altera FPGA include look-up table logic elements, registers, memory blocks and DSP elements that can be broken down into 18-bit by 18-bit multipliers. The required FPGA resources are reduced by filter optimization techniques, time-sharing of multipliers and reduction of multipliers through the use of successive approximation circuits where suitable. Although the use of successive approximation circuits increases the use of registers, look-up table and memory elements over multiplier based implementations, these circuits are examined as an alternative to the use of multipliers due to the large number of these logic elements available on FPGA devices relative to the available multipliers.

It is noted that structures for QAM MODEMs have previously been developed using ASIC implementations such as the MAX5880 and STV0297J chips [4, 30]. Though, given the resources required to develop ASIC devices, these implementations have been restricted to larger companies and generally have been proprietary to these companies. FPGAs now make it possible for smaller companies with reduced resources, to implement these systems. This work is intended to allow these smaller companies to implement QAM systems using FPGAs without the need for expensive ASIC designs.

1.6 Thesis Outline

Quadrature amplitude modulation (QAM) is reviewed in Chapter 2. An overview of the upstream channel as defined in DOCSIS 3.0 is given in Chapter 3. This chapter includes some communication theory necessary for the understanding of the upstream channel as well as impairments of interest that are encountered throughout the upstream channel.

Chapter 4 outlines some practical implementation constraints placed upon the example design of the digital front end as well as the DOCSIS constraints imposed upon the design.

Chapter 5 explains how the performance of the designed receiver will be measured as well as design methods, and the details about the theoretical design of the modules required for the example implementation.

Chapter 6 details the practical implementation of the modules that were theoretically designed and simulated in Chapter 5. This chapter covers bit sizing of the finite precision circuits created, as well as testing of the individual circuits and the testing of the digital front end as a whole.

Finally, Chapter 7 concludes the work presented in this thesis by giving an end-to-end performance measurement of the front-end as well as resource utilization figures. In addition this chapter brings to light future work that could be done to improve the feature set of the digital front end implementation presented here.

Chapter 2

QAM System Description

Understanding the development of FPGA structures for a DOCSIS 3.0 QAM demodulator, as proposed in this thesis, requires an understanding of the operation of a QAM system.

The following section provides an introduction to discrete-time QAM systems. The basic QAM system described is not the most efficient structure for implementation, but it does provide a suitable base for later chapters.

2.1 Introduction to QAM

A basic QAM system consists of a modulator and a demodulator and these are described in the following two subsections.

2.1.1 QAM Modulator

A QAM modulator maps binary data at its input, taken l bits at a time, into one of $s_m(t)$ signals, where $1 \leq m \leq M$, $M = 2^l$ ($l = \log_2 M$). A QAM modulation scheme, where there are M possible transmit signals, is referred to as M-QAM. For example, if $l = 4$, then $M = 16$ and the system is referred to as 16-QAM, which maps the 16 possible 4-bit sequences into one of 16 possible signals.

The modulator outputs a new signal, $s_m(t)$, every T seconds to produce the signal, $s(t)$, which is transmitted over the channel (a coax cable). Thus, in each second $R = 1/T$ of the $s_m(t)$ signals are transmitted. R is the signalling rate. It is also referred to as the symbol rate, since in the modulator the l bits are first mapped into one of M symbols, prior to generating the signal $s_m(t)$. Similarly, since each $s_m(t)$ signal is mapped from l bits, the bit time is given by $T_b = T/l$ and the bit rate is $R_b = lR$ bits/second.

A block diagram of a basic discrete-time QAM modulator using a discrete-time representation is shown in Figure 2.1.

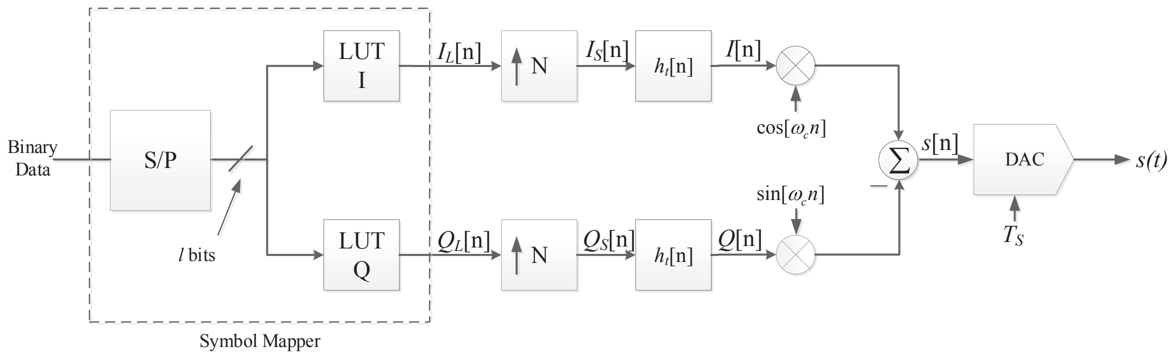


Figure 2.1: Discrete-time QAM modulator

The modulator in Figure 2.1 generates the discrete-time signal $s[n]$, which is passed through a DAC (digital to analog converter), with a sampling rate of $R_s = 1/T_s$, where T_s is the sampling period, to produce the continuous-time signal $s(t)$. $s(t)$ can be viewed as the sum of two amplitude modulated signals, one generated in the upper path of Figure 2.1 using a cosine carrier and the other generated in the lower path of the figure using a sine carrier. These two carriers, referred to as quadrature carriers, are 90 degrees out of phase and thus are orthogonal. Since they are orthogonal, the two modulated carriers in the sum do not interfere with each other. The upper path in Figure 2.1, with the cosine carrier, is typically referred to as the in-phase path and the lower path, with the sine carrier, is referred to as the quadrature path.

The input to the modulator block diagram in Figure 2.1 is binary data, which is a serial bit sequence. The S/P block in the Bit to Symbol Mapper is a serial to parallel converter that partitions the input bit sequence into non-overlapping segments of $l = \log_2 M$ bits. The l bit output of the S/P block is an address used by the LUT (Look Up Table) blocks to locate the symbol amplitude values to be used in the in-phase and quadrature paths. The amplitude, for the k^{th} symbol, at the output of the in-phase look up table, LUT I, is represented by $a_I[k]$ and the amplitude at the output of the quadrature look up table, LUT Q, is given by $a_Q[k]$, where k is the symbol index. These two amplitude values, given as an ordered pair ($a_I[k]$,$a_Q[k]$), represent a QAM symbol. It is useful to represent this ordered pair, ($a_I[k]$,$a_Q[k]$), which is a two-dimensional representation, as a complex number, $a_I[k] + ja_Q[k]$, which is also a two-dimensional representation. An example of a 16-QAM look up table mapping, using the more compact complex number representation, is given in Table 2.1.

Table 2.1: 16-QAM look up table mapping using complex notation

l Bit Input	Amplitude $(a_I[k] + ja_Q[k])$
0000	-3-j3
0001	-3-j1
0010	-3+j3
0011	-3+j1
0100	-1-j3
0101	-1-j1
0110	-1+j3
0111	-1+j1
1000	+3-j3
1001	+3-j1
1010	+3+j3
1011	+3+j1
1100	+1-j3
1101	+1-j1
1110	+1+j3
1111	+1+j1

A very useful visual interpretation of all possible QAM symbols is given by a constellation diagram, which is obtained by plotting all M possible symbols. An example constellation diagram for 16-QAM is given in Figure 2.2, which is a plot of all of the symbols in Table 2.1.

Mathematically, the output of the look up table blocks could be viewed as a sequence of weighted impulses, where $\delta[\cdot]$ is a Kronecker delta:

$$I_L[n] = \sum_k a_I[k]\delta[n-k]; \tag{2.1}$$

$$Q_L[n] = \sum_k a_Q[k]\delta[n-k]. \tag{2.2}$$

Following the look up tables in Figure 2.1, are the upsampling blocks, labelled $\uparrow N$, and the pulse shaping filters, where $h_t[n]$ is the discrete-time impulse response of the pulse shaping filter. These two blocks are used to generate an amplitude weighted pulse for each of the input symbols. In order to generate the desired symbol rate, R, at the output of the

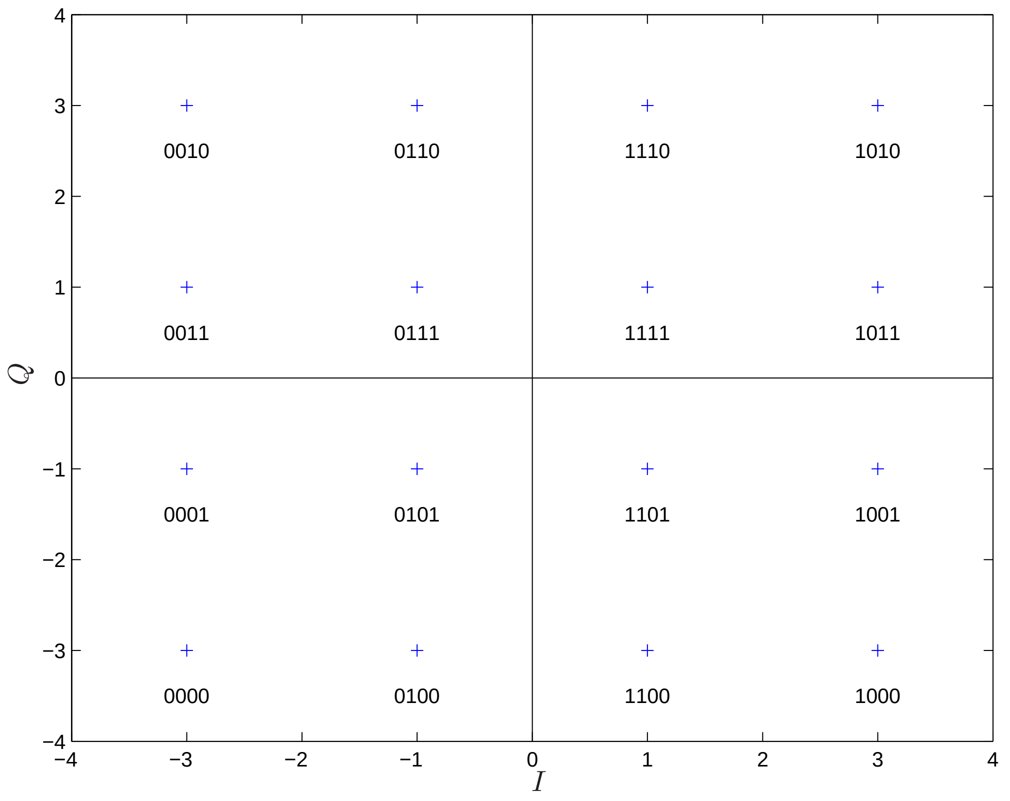


Figure 2.2: Constellation diagram for 16-QAM

DAC, the value of N must be the ratio of the sampling rate to the symbol rate,

$$N = R_s/R = T/T_s$$

.

The upsampler simply inserts $N-1$ zeros in between each symbol amplitude value, a process known as zero-stuffing, which mathematically is represented as

$$I_s[n] = \begin{cases} I_L[n/N] & n = 0, N, 2N, \cdots \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

for the in-phase upsampling block. Substituting (2.1) into (2.3) gives

$$\begin{aligned} I_S[n] &= \sum_k a_I[k]\delta[n/N-k] \\ &= \sum_k a_I[k]\delta[n-kN]. \end{aligned} \tag{2.4}$$

Similarly, for the quadrature path,

$$Q_S[n] = \sum_k a_Q[k]\delta[n-kN]. \tag{2.5}$$

The pulse shaping filters are lowpass filters that are, in part, designed to remove the spectral images produced by the upsampling operation and to limit the bandwidth of the transmitted signal. The output of the pulse shaping filters is the convolution of the filter input with the impulse response of the filter, $h_t[n]$, which results in

$$I[n] = I_S[n] * h_t[n] = \sum_k a_I[k]h_t[n-kN] \tag{2.6}$$

$$Q[n] = Q_S[n] * h_t[n] = \sum_k a_Q[k]h_t[n-kN], \tag{2.7}$$

where * represents convolution.

The outputs of the filters are multiplied by the cosine and sine carriers and the results are then subtracted to give

$$s[n] = I[n]\cos(\omega_c n) - Q[n]\sin(\omega_c n), \tag{2.8}$$

where ω_c is the carrier frequency with units radians/sample. The continuous-time output of the DAC, which is the RF (radio frequency) signal for transmission, is

$$s(t) = I(t)\cos(\Omega_c t) - Q(t)\sin(\Omega_c t), \tag{2.9}$$

where $\Omega_c = \omega_c T_s$ is the carrier frequency with units radians/second, and

$$I(t) = \sum_k a_I[k] h_t(t - kT) \tag{2.10}$$

$$Q(t) = \sum_k a_Q[k] h_t(t - kT), \tag{2.11}$$

where $h_t(t)$ is the continuous-time impulse response of the pulse shaping filter.

2.1.2 QAM Demodulator

The basic function of a QAM demodulator is to recover the binary data that were used to generate the signal $s(t)$ in Figure 2.1. This task is challenging, since the received signal, $\tilde{s}(t)$, after travelling through the transmission channel is usually corrupted by noise.

A block diagram of a basic QAM demodulator using a discrete-time representation is shown in Figure 2.3.

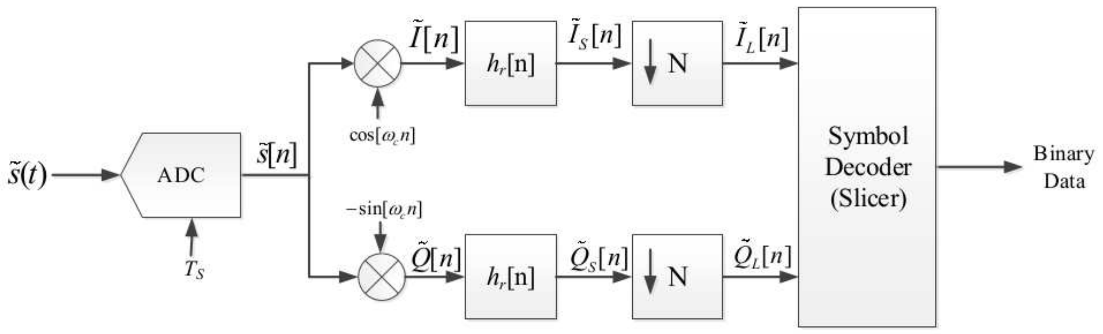


Figure 2.3: Discrete-time QAM Demodulator

The input to the demodulator is the signal $\tilde{s}(t) = s(t) + w(t)$, which is a corrupted version of $s(t)$, where $w(t)$ is additive noise. Though, initially, to simplify the description of the operation of the demodulator, it is assumed that $\tilde{s}(t) = s(t)$. The continuous-time signal $\tilde{s}(t)$ is converted to the discrete-time signal $\tilde{s}[n]$ by the ADC (analog to digital converter). The next step is to downconvert the RF signal to baseband by multiplying the signal $\tilde{s}[n]$ by the two quadrature carriers generated in the demodulator. The in-phase component is

$$\begin{aligned}\tilde{I}[n] &= \tilde{s}[n]\cos(\omega_c n) \\ &= I[n]\cos(\omega_c n)\cos(\omega_c n) - Q[n]\sin(\omega_c n)\cos(\omega_c n) \\ &= \frac{1}{2}I[n](\cos(2\omega_c n) + \cos(0)) - \frac{1}{2}Q[n](\sin(2\omega_c n) + \sin(0)) \\ &= \frac{1}{2}I[n] + \frac{1}{2}I[n]\cos(2\omega_c n) - \frac{1}{2}Q[n]\sin(2\omega_c n), \qquad (2.12)\end{aligned}$$

where (2.8) and $\tilde{s}[n] = s[n]$ are used, and similarly the quadrature component is

$$\begin{aligned}\tilde{Q}[n] &= -\tilde{s}[n]\sin(\omega_c n) \\ &= -I[n]\cos(\omega_c n)\sin(\omega_c n) + Q[n]\sin(\omega_c n)\sin(\omega_c n) \\ &= -\frac{1}{2}I[n](\sin(2\omega_c n) + \sin(0)) + \frac{1}{2}Q[n](-\cos(2\omega_c n) + \cos(0)) \\ &= \frac{1}{2}Q[n] - \frac{1}{2}I[n]\sin(2\omega_c n) - \frac{1}{2}Q[n]\cos(2\omega_c n). \qquad (2.13)\end{aligned}$$

The result of the down-conversion operation, as shown in equations (2.12) and (2.13), is the scaled baseband signals corrupted by double frequency sinusoidal terms. These double frequency terms are removed by the low pass filter with impulse response $h_r[n]$. $h_r[n]$ is typically chosen to be the time reversed version of the pulse shaping filter impulse response, $h_t[n] = h_r[-n]$, and is called a matched filter. A matched filter gives the maximum signal to noise ratio when $w[n]$ is additive white Gaussian noise. If the pulse shaping filter impulse response has even symmetry, which is generally the case, then the impulse responses of the matched filter and pulse shaping filter are identical and

$$h_t[n] = h_r[n] = h[n]. \qquad (2.14)$$

The output of the in-phase matched filter is

$$\tilde{I}_S[n] = \tilde{I}[n] * h_r[n] + v_I[n], \qquad (2.15)$$

where a noise term, $v_I[n]$, has been included. $v_I[n]$ is the noise at the output of the matched filter due to the additive noise $w[n]$ in $\tilde{s}[n] = s[n] + w[n]$ at the input to the demodulator. Substituting (2.6) and (2.12) into (2.16), noting that the double frequency terms in (2.12) are filtered out, gives,

$$\begin{aligned}
\tilde{I}_S[n] &= \frac{1}{2} I_S[n] * h_t[n] * h_r[n] + v_I[n] \\
&= \frac{1}{2} I_S[n] * r_p[n] + v_I[n] \\
&= \frac{1}{2} \sum_k a_I[k] r_p[n - kN] + v_I[n].
\end{aligned} \tag{2.16}$$

where the convolution of the impulse responses of the pulse shaping filter and the matched filter gives the combined impulse $g[n] = h_t[n] * h_r[n]$. This combined impulse response is equivalent to the pulse shaping autocorrelation function, $r_p[n] = h_t[n] * h_t[-n]$, since the impulse response of the matched filter is a time reversed version of the pulse shaping filter. Similarly, for the quadrature path

$$\begin{aligned}
\tilde{Q}_S[n] &= \tilde{Q}[n] * h_r[n] + v_Q[n] \\
&= \frac{1}{2} \sum_k a_Q[k] r_p[n - kN] + v_Q[n].
\end{aligned} \tag{2.17}$$

The next step is to determine the symbol values, which can be recovered by selecting every Nth sample in (2.16) and (2.17). This is implemented in the demodulator block diagram using a downsampler, which is the complimentary operation to the upsampler used in the modulator. The downsampling block in Figure 2.3, is labelled $\downarrow N$. The downsampler simply keeps every Nth sample, discarding the other $N - 1$ in-between samples. The downsample operation is mathematically represented as

$$\tilde{I}_L[n] = \tilde{I}_S[nN] \tag{2.18}$$

for the in-phase downsampling block. Substituting (2.16) into (2.18) and assuming the pulse,

$h_t[n]$, has unit energy (i.e. $r_p[0] = 1$) gives

$$\begin{aligned}\tilde{I}_L[n] &= \frac{1}{2}\sum_k a_I[k]r_p[nN - kN] + v_I[nN] \\ &= \frac{1}{2}a_I[n] + \frac{1}{2}\sum_{k\neq n} a_I[k]r_p[(n-k)N] + v_I[nN]. \qquad (2.19)\end{aligned}$$

Similarly, for the quadrature path,

$$\tilde{Q}_L[n] = \frac{1}{2}a_Q[n] + \frac{1}{2}\sum_{k\neq n} a_Q[k]r_p[(n-k)N] + v_Q[nN]. \qquad (2.20)$$

In equations (2.19) and (2.20), if, for the moment, it is assumed that $h_t[n]$ has a finite length of N samples, the second term (i.e. the summation term) in the equations will be 0. Also, the amplitude values in these equations have been multiplied by 2 to scale them to the values used in the modulator, to give

$$\tilde{I}_L[n] = a_I[n] + v_I[nN] \qquad (2.21)$$

$$\tilde{Q}_L[n] = a_Q[n] + v_Q[nN]. \qquad (2.22)$$

The additive noise in (2.21) and (2.22) will move the constellation point ($\tilde{I}_L[n]$, $\tilde{Q}_L[n]$), for example one of the constellation points in Figure 2.2, away from the true value ($a_I[n]$, $a_Q[n]$). An error occurs when the noise moves ($\tilde{I}_L[n]$, $\tilde{Q}_L[n]$) closer to a constellation point that was not the one that was sent, though in an acceptable noise environment, the correct constellation point will most likely be selected.

The first term of equations (2.21) and (2.22) can also be represented as a sequence of weighted impulses,

$$\tilde{I}_L[n] = \sum_k a_I[k]\delta[n-k] + v_I[nN] \qquad (2.23)$$

$$\tilde{Q}_L[n] = \sum_k a_Q[k]\delta[n-k] + v_Q[nN], \qquad (2.24)$$

which is the original set of transmitted symbols in equations (2.1) and (2.2) corrupted by noise.

The function of the Symbol Decoder, also known as a slicer, in Figure 2.3 is to choose the most likely symbol constellation point and then perform the complimentary operation of the Bit to Symbol Mapper, in Figure 2.1, to obtain the original l binary bits for the symbol.

The previous assumption that $h_t[n]$ has a finite length of N samples, is not accurate because the pulse shaping filter is a low pass filter designed in part to limit the bandwidth of the transmitted QAM signal. This bandlimiting has the effect of spreading out the pulse in time. If the duration of each pulse is greater than the symbol period, T, then the symbol pulses can interfere with each other. This interference is referred to as intersymbol interference (ISI). The summation terms in equations (2.19) and (2.20) are the ISI terms due to pulse spreading. This ISI could be prevented if the combined impulse response of the pulse shaping and matched filters, $g[n]$ or equivalently $r_p[n]$, is zero at multiples of the symbol period, N,

$$g[nN] = r_p[nN] = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases} . \tag{2.25}$$

If $g[n]$, which is of length K symbols (KN samples), where K is a positive integer, satisfies the Nyquist criterion of (2.25), then the ISI term of equation (2.19) reduces to 0,

$$\frac{1}{2}\sum_{k\neq n} a_I[k] r_p[(n-k)N] = 0, \tag{2.26}$$

and similarly for equation (2.20),

$$\frac{1}{2}\sum_{k\neq n} a_Q[k] r_p[(n-k)N] = 0. \tag{2.27}$$

Since the ISI terms are 0, the $g[n]$ pulses for each of the symbols will not interfere with each other at the sampling points, kN. Recall that the symbol sampling is implemented as the downsampler in Figure 2.3.

Filters designed to meet equation (2.25) are said to meet the Nyquist criterion. One such filter is the raised cosine filter, which derives its name from its magnitude response, which has a cosine shaped transition band. The discrete-time impulse response of a raised cosine filter is given by,

$$g[n] = \begin{cases} \frac{\pi}{4N}\text{sinc}\left(\frac{1}{2\beta}\right) & n = \pm\frac{N}{2\beta} \\ \frac{1}{N}\text{sinc}\left(\frac{n}{N}\right)\frac{\cos\left(\frac{\pi\beta n}{N}\right)}{1-\left(\frac{2\beta n}{N}\right)^2} & \text{otherwise} \end{cases} \tag{2.28}$$

where $0 \leq \beta \leq 1$ is the roll-off parameter, which defines the transition bandwidth of the filter.

Figure 2.4 gives a demonstration of the Nyquist criterion for the raised cosine pulse. The continuous-time representation of the raised cosine pulse is used to better demonstrate the Nyquist criterion. The figure shows a sequence of three raised cosine symbols. In this figure, each of the raised cosine waveforms is zero at all symbol time samples, except at the peak of the waveforms. Thus, there is no ISI between the symbols when sampled at the symbol rate $(1/T)$.

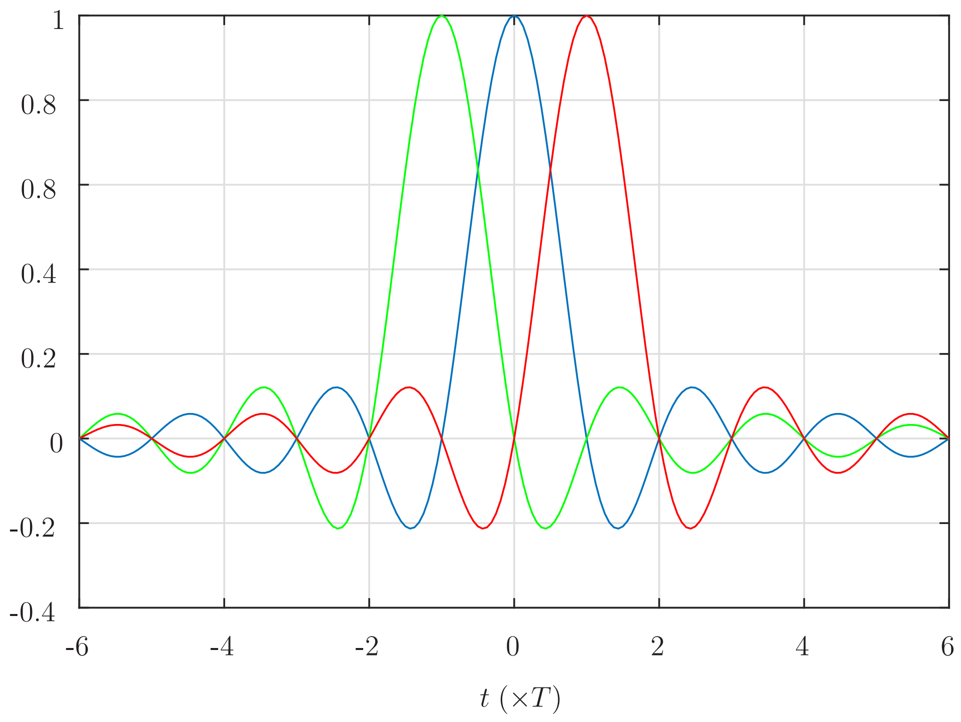


Figure 2.4: Demonstration of Nyquist criterion for raised cosine symbol pulses

Recall that $g[n]$ is the convolution of the transmit pulse shaping filter and the receive matched filter, $g[n] = h_t[n] * h_r[n] = h[n] * h[n]$. The two filters have the same form $h[n]$.

A filter that meets the requirement that the convolution of the filter with itself results in a raised cosine filter is the square root raised cosine (SRRC) filter, which will be discussed in further detail in the following chapter. The SRRC filter is used in this thesis for the pulse shaping and matched filters.

Chapter 3
The Upstream Channel

3.1 DOCSIS Upstream Channel Overview

CATV networks have evolved to provide better on-demand and data services to customers, although the upstream portion of the CATV network and associated standards have been evolving at a higher rate than that of the downstream. These changes have been made to allow customers to send more data over the CATV structure to allow faster communication and request data at faster speeds. The primary focus of the research presented in this thesis is based upon the upstream portion of the DOCSIS 3.0 standard.

The upstream has been confined to the lower portions of the frequency spectrum available to the CATV network and as such is very susceptible to low frequency impairments in the upstream communications channel. The signal being transmitted through the upstream channel will likely encounter delay, noise and other impairments resulting in frequency and phase shifts. These impairments must therefore be compensated for when designing an upstream demodulator. The following sections in this chapter provide background on the theory behind the upstream channel and the impairments that must be compensated for as well as specifications for the upstream channel as defined in the DOCSIS 3.0.

3.1.1 DOCSIS 3.0 Physical Layer Specifications

The DOCSIS specifies that the symbol rate of a transmission in the upstream channel can be in the range of 160 to 5120 ksym/s and must have a channel bandwidth that may not exceed 6.4 MHz [7].

In a DOCSIS based system there are modulators and demodulators at both the headend

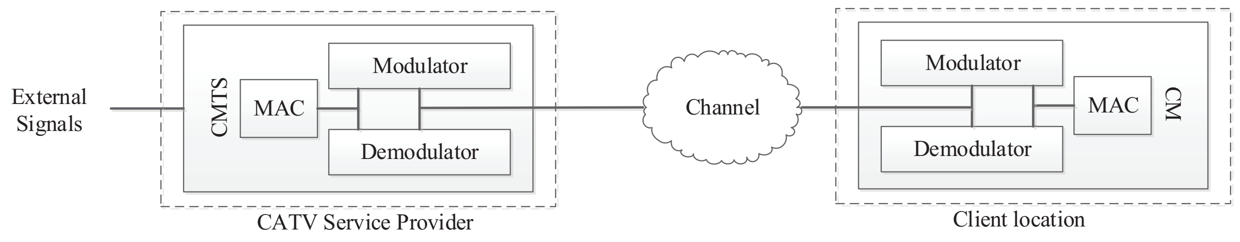


Figure 3.1: CMTS and CM in a CATV Network

(CMTS) and client side (CM) locations, as shown in figure 3.1, to allow for bi-directional communication. Communication over the CATV network can be done using either (time division multiple access) TDMA or synchronous code division multiple access (S-CDMA) bursts. The DOCSIS specifies that all transmissions must use a quadrature amplitude modulation (QAM) scheme.

It is the job of the modulator to take the binary data signals and convert them to radio frequency (RF) signals to be transmitted over the CATV network. The demodulator has the opposite job. It must read the RF signals and correct for any attenuation caused during transmission by using an automatic gain control circuit (AGC) then digitally sample the signal using an analog to digital converter (ADC). After the signal has been gain corrected and digitally sampled the samples are then passed off to the digital front-end of the demodulator for further processing. Once the front-end has completed processing an equalizer is used to correct for further impairments that cannot be corrected in the digital front-end.

The CMTS headend unit uses a Media Access Controller (MAC) to keep track of when data bursts should be sent and on what frequencies the bursts should be transmitted to client units. The MAC located in the CM unit also keeps track of when the CM can send data back to the CMTS and on what frequencies these transmissions should occur.

The DOCSIS 3.0 specification used throughout this thesis defines the data to be transmitted upstream to be sent as a QAM signal using TDMA burst packets. The transmitted data packet must consist of a QPSK preamble that can be up to 1536 bits in length, followed by QAM data of any of the supported QAM schemes. The format of the data packet is shown in figure 3.2.

The preamble in the data packet can be used to detect the start of a packet transmission as well as for the correction of signal impairments that occur after transmission of the packet and

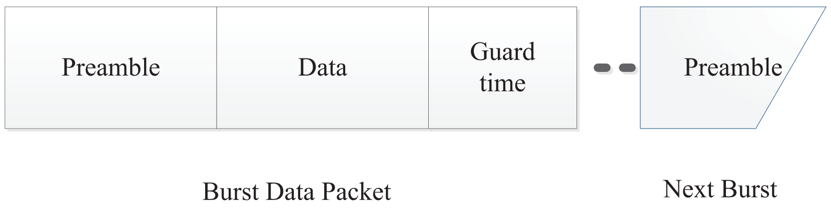


Figure 3.2: DOCSIS Burst Data Packet High Level Structure

before receiving the packet at the demodulator. The data transmissions are to be separated by a guard time prior to the transmission of the next data packet in TDMA mode. This guard time gives the demodulator time to reconfigure for the next transmission if necessary.

3.1.2 Pulse Shaping and Matched Filtering

The previous chapter discussed the use of square root raised cosine (SRRC) filters for the pulse shaping filter in the transmitter and for the matched filter in the receiver. This section further examines the SRRC filter. The continuous-time impulse response and frequency response for the SRRC filter are given in equations (3.1) and (3.2), respectively:

$$h_{\text{SRRC}}(t) = \begin{cases} \frac{1}{T}\left(1+\beta(\frac{4}{\pi}-1)\right) & ,t=0 \\ \frac{\beta}{T\sqrt{2}}\left[\left(1+\frac{2}{\pi}\right)sin\left(\frac{\pi}{4\beta}\right)+\left(1-\frac{2}{\pi}\right)cos\left(\frac{\pi}{4\beta}\right)\right] & ,t=\pm\frac{T}{4\beta} \\ \frac{1}{T}\frac{sin\left[\pi\frac{t}{T}(1-\beta)\right]+4\beta\frac{t}{T}cos\left[\pi\frac{t}{T}(1+\beta)\right]}{\pi\frac{t}{T}\left[1-\left(4\beta\frac{t}{T}\right)^2\right]} & ,\text{otherwise} \end{cases} \tag{3.1}$$

$$H_{\text{SRRC}}(f) = \begin{cases} \sqrt{T} & ,|f|\le\frac{1-\beta}{2T} \\ \sqrt{T\left[cos^2\left(\frac{\pi T}{2\beta}\left[|f|-\frac{1-\beta}{2T}\right]\right)\right]} & ,\frac{1-\beta}{2T}<|f|\le\frac{1+\beta}{2T} \end{cases} \tag{3.2}$$

where t is time in seconds, T is the symbol period in seconds, f is the frequency being evaluated in Hertz and β is the roll-off factor of the filter.

The roll-off factor, β, of the SRRC filter has the effect of changing the bandwidth of the filter and the amount of attenuation achieved in the stop band of the filter. The bandwidth (BW) of the filter centered at DC can be determined based upon the symbol frequency, F_{sym}, and the roll-off factor using equation (3.3). The effect of the roll-off factor is plotted in figure

3.3 for clarity in understanding the effects of changing β. The figure shows a SRRC filter that is 65 coefficients long with various roll-off factors. The distance between the dashed lines of the same colours in the figure indicate the bandwidth of the SRRC filter for each roll-off factor.

$$BW = F_{sym}(1+\beta) \tag{3.3}$$

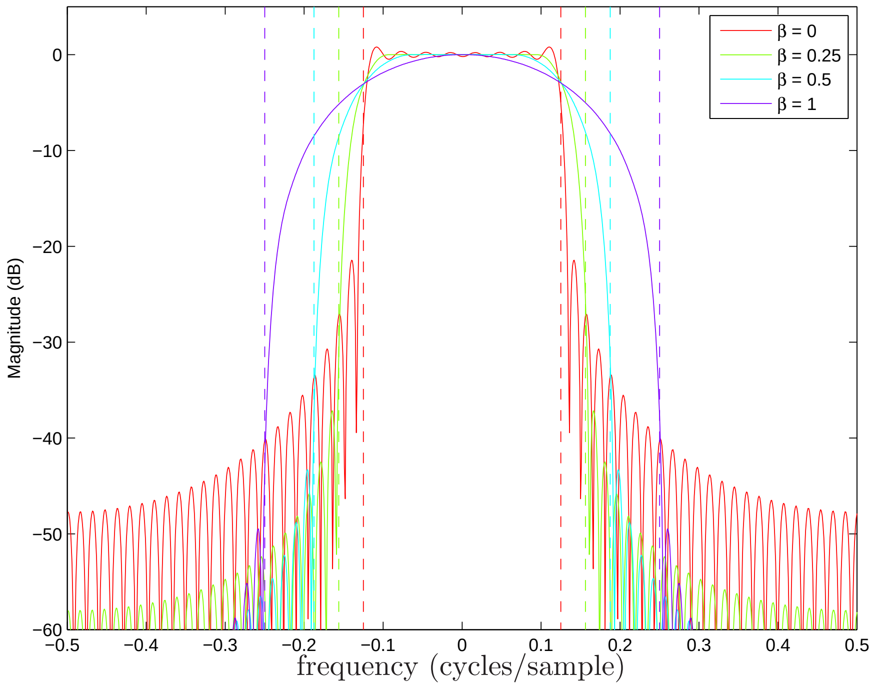


Figure 3.3: The Effect of the Rolloff Factor of a SRRC Filter

The DOCSIS has different requirements for the transmitting pulse-shaping filter and the receiver's matched filter. There are strict requirements on out-of-band emissions for the transmitting filter to ensure that the transmitted signal's spectrum does not interfere with channel transmissions in close spectral proximity (referred to as adjacent channels). While the DOCSIS states that the transmitting filter must approximate a SRRC filter with a roll-off factor, β, of 0.25, the filter can be windowed (applying weighting to the filter coefficients) and have its rolloff factor adjusted to meet out-of-band specifications.

The receiver's SRRC matched filter does not have these strict emission requirements, however is specified to have a β of 0.25 and therefore its attenuation is based primarily upon filter length.

Although theoretically the response of the RC filter provides no interference between adjacent symbols, in practice this non-interference cannot be realized as discussed in the following subsection.

3.1.3 Inter-symbol Interference

As discussed in the previous chapter, the impulse response of a RC filter (or cascade of two SRRC filters) has zero-crossings at symbol times resulting in each filtered symbol causing no interference with the previous or subsequent symbols. Unfortunately, to achieve this property of the RC filter, the filter must be infinitely long and as such cannot be practically implemented.

As the number of coefficients in each of the two SRRC filters needed in the transmitter and receiver circuits becomes reduced, the zero-crossings of the filters' cascaded impulse response no longer occur at the symbol times. These non-zero symbol time points add into the adjacent transmitted symbols and cause interference referred to as inter-symbol interference (ISI). Figure 3.4 shows the amount of ISI produced at each symbol time for three different SRRC filters of length 32, 24 and 16 symbols. It can be seen from figure 3.4 that as the filter's length is reduced the ISI tends to increase.

The amount of interference that leaks into adjacent symbols will have a detrimental effect upon the reconstructed data at the end of the receiver as the amplitude of each transmitted symbol will be affected by previously transmitted symbols.

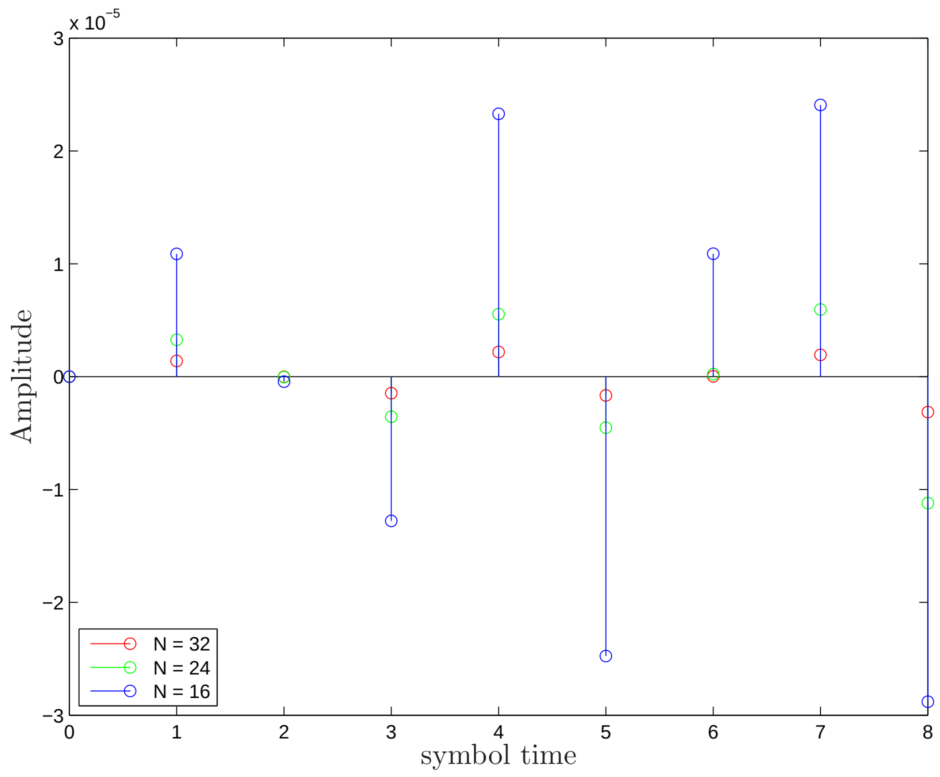


Figure 3.4: ISI Amplitude for Various Length SRRC Filters

3.2 Theory Behind the End-to-End System

To achieve a better understanding of how the channel impairments affect the overall system performance it is beneficial to first understand the end-to-end DOCSIS communications system. This section will examine the theory behind the transmission and reception of the signals being sent through the DOCSIS channel.

3.2.1 The Transmitter

A standard DOCSIS 3.0 transmitter is shown in figure 3.5, which is based upon the QAM modulator discussed in the previous chapter.

Binary data are first mapped to digital amplitude values through the use of a mapper as described in the previous chapter. Once the binary data have been converted to discrete amplitude pulses these pulses are upsampled by zero stuffing. These additional zero samples give the appearance of a higher sampling rate, as now several additional samples occur prior

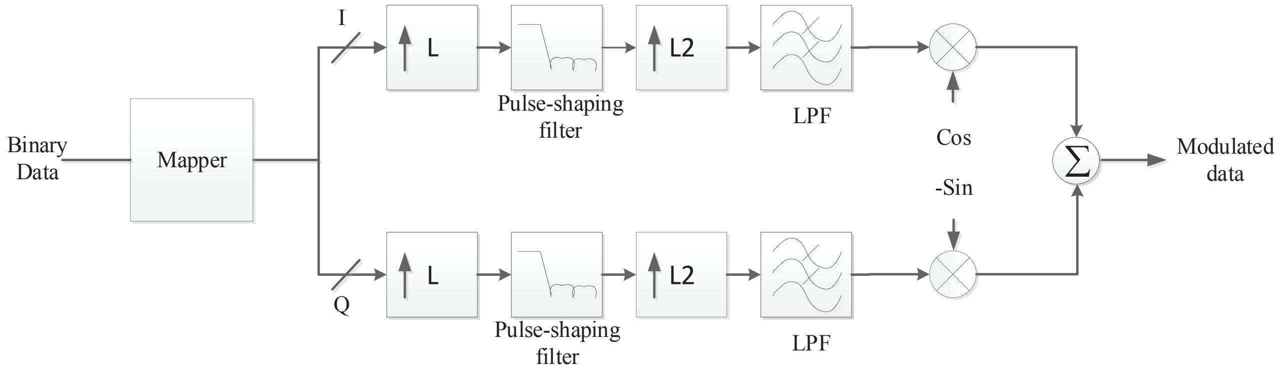


Figure 3.5: DOCSIS 3.0 Modulator Block Diagram

to the actual mapped samples, and allow for the clocks in the system to now be run at a higher rate as well.

Zero-stuffing creates additional spectral images located every $\frac{F_s}{L}$ Hz from DC to F_s Hz, where F_s is the sampling rate of the upsampled signal. These images distort the original discrete pulses and must be filtered out to restore the original signal. Figures 3.6 and 3.7 show a RC frequency response and the images created from up-sampling the RC filter with zero-stuffing by a factor of 4, respectively.

The effects of zero-stuffing upsampling can be mitigated by the use of a SRRC filter. The SRRC filter, as discussed earlier, is a special type of low-pass filter that works well for reducing the bandwidth requirements of pulses and is also able to filter the undesired upsampling images as long as the SRRC is designed to have the same number of samples per symbol as the amount by which the data pulses have been upsampled. Figure 3.8 shows the frequency response of a SRRC filter that has been designed for four samples per symbol at its input overlaid upon the upsampled frequency spectrum in figure 3.7. It can be easily seen that this filter will remove the images created due to the up-sampling of the discrete amplitude pulses.

Once the SRRC filtering has been completed there is generally a second stage of upsampling that occurs to increase the sampling rate of the SRRC filter output to a higher frequency that allows for an RF transmission. This additional upsampling in turn requires another low pass filter to remove upsampling images.

The filtered upsampled pulses are then each multiplied by a carrier frequency out of phase

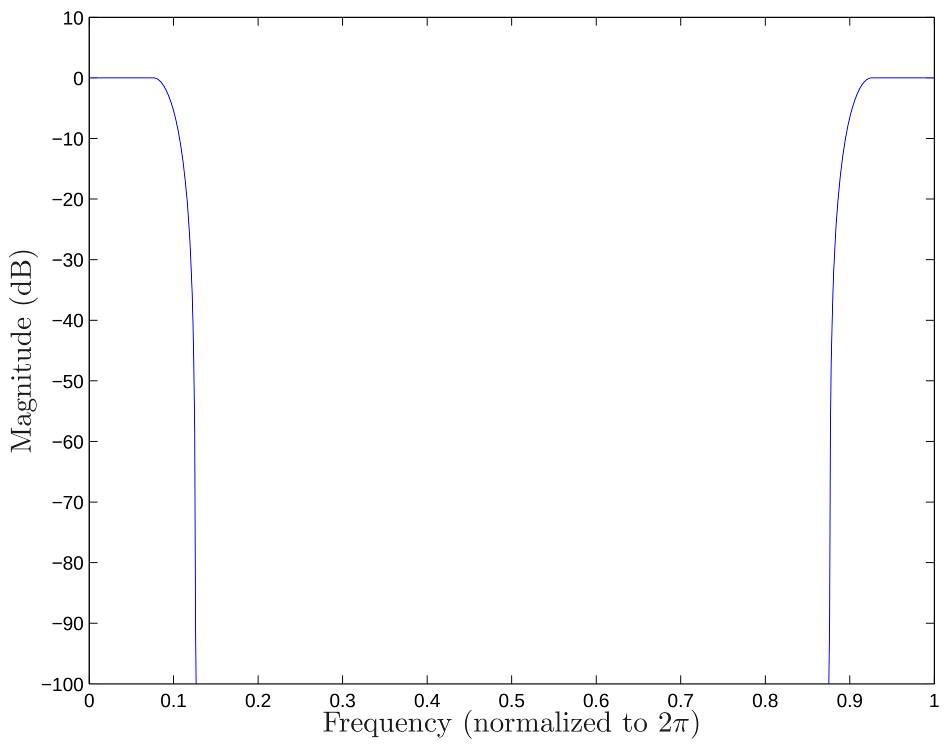


Figure 3.6: RC Frequency Response

by $\frac{\pi}{2}$ rad from one another producing in-phase (I) and quadrature phase (Q) signals. The multiplication of the carrier frequency causes the baseband (DC centered) signal to be shifted in frequency to that of the carriers. The resulting signals are then summed and transmitted as a QAM signal. The resulting equation for the transmitted signal is given in (3.4),

$$s_{tx}[n] = I[n]Cos(\omega_0 n) - Q[n]Sin(\omega_0 n), \tag{3.4}$$

where ω_0 is the frequency of the transmit carrier signals in rad/sample.

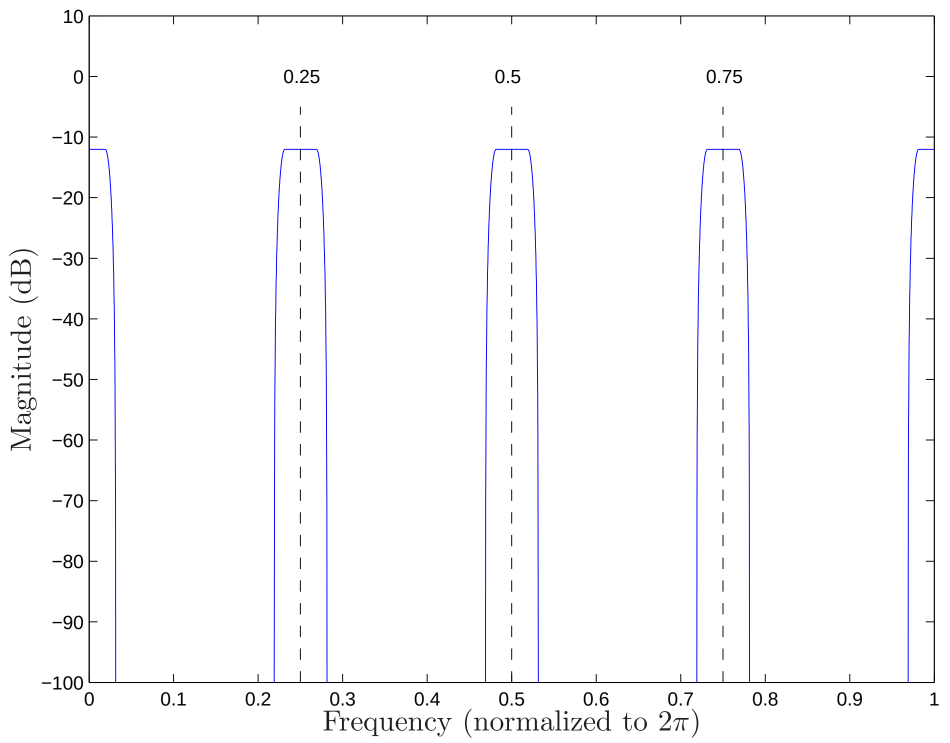


Figure 3.7: Images created from upsampling by 4 with zero-stuffing

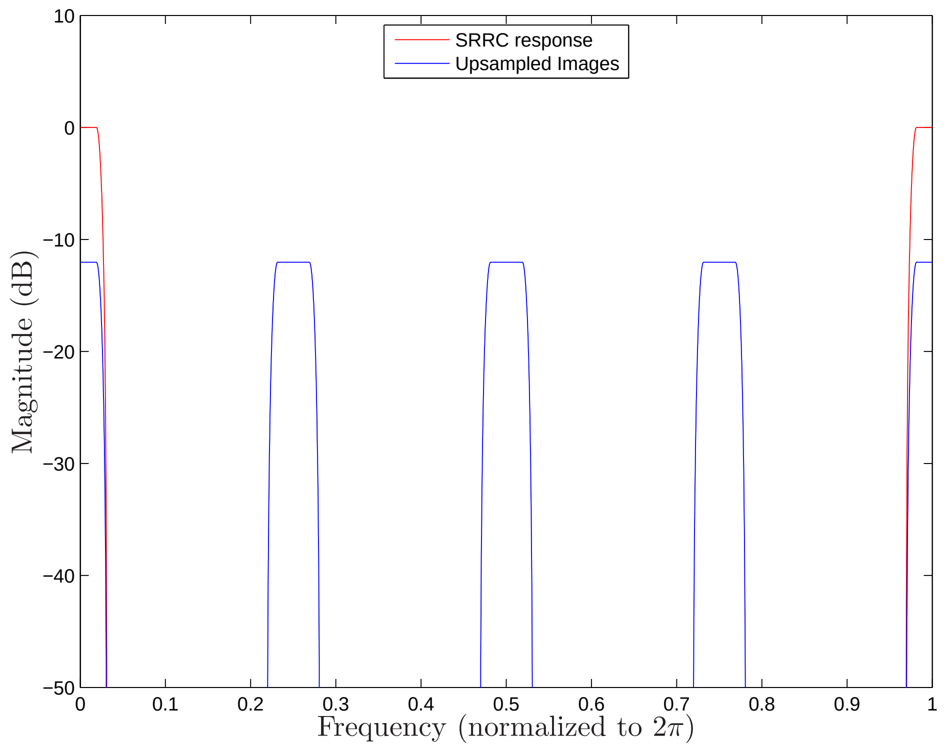


Figure 3.8: SRRC with 4 samples per symbol overlaid with images caused by upsampling by 4

3.2.2 The Demodulator Digital Front End

The demodulator, also known as a receiver, of a typical DOCSIS 3.0 compliant system is shown in figure 3.9 and is based upon the QAM demodulator discussed in chapter 2.

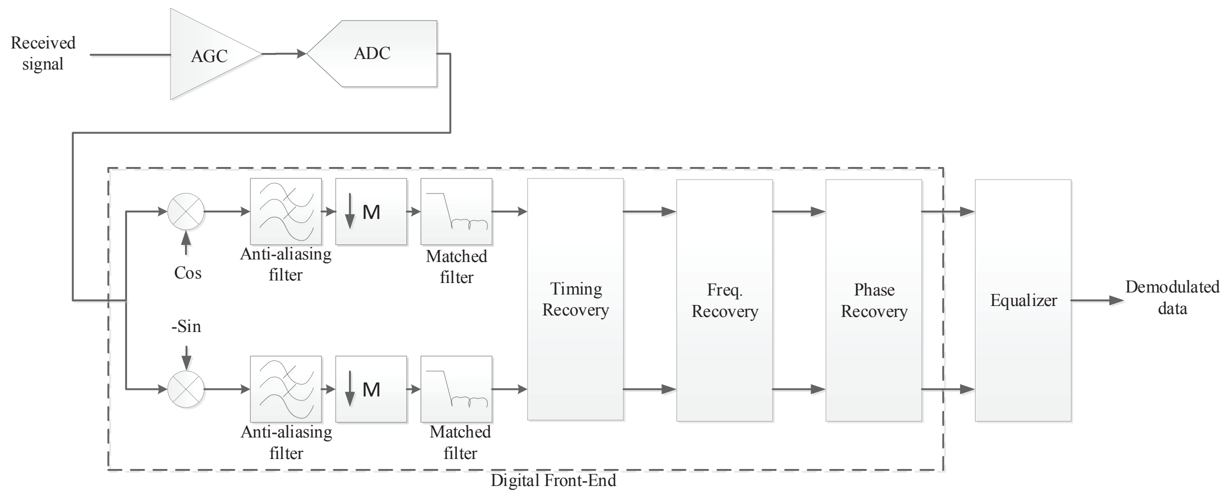


Figure 3.9: DOCSIS 3.0 Demodulator Block Diagram

The first step in the digital front-end of the demodulator is to down-convert the transmitted QAM signal back to base-band by the multiplication of a sine and cosine of the desired down-conversion frequency. Once this down-conversion has completed the resulting double frequency components are removed by a low pass filter.

The demodulator then downsamples the signal in order to reduce the down-converted I and Q signals back to the sampling rate required by the matched filter. The downsampling is performed by discarding equally time spaced samples from the received sampled data. When a signal is downsampled its bandwidth relative to the new sampling frequency is increased by the amount by which it is being downsampled. If any portion of the received signal spectrum, prior to downsampling, exceeds half of the downsampled signal's sampling frequency it is aliased (leaks back into) into the downsampled signal. To reduce the effects of aliasing caused by downsampling, an anti-aliasing filter is required.

This filter must have a stop-band corner frequency that begins before the frequency given by (3.5) in order to stop aliasing of the signals about to be downsampled,

$$F_{stop} = \frac{F_s}{M}, \tag{3.5}$$

where F_s is the sample frequency of the system and M is the downsampling factor.

The signal is then passed through a matched filter to restore the approximate RC filter response required to reduce ISI and reconstruct the original transmitted binary data.

The following sections describe the theory need for the timing, frequency and phase synchronization that will be implemented.

3.2.3 Timing Impairments

Delays through the upstream channel as well as hardware mismatches between the transmitter and receiver units can cause timing impairments in the form of incorrect sampling times at the receiver.

In order to correctly decode the transmitted data the receiver must sample the incoming signal, once down-converted and filtered, at the correct time.

Figure 3.10 shows the effect of a single sample offset at the output of a matched filter designed to operate at four samples per symbol. It is clear from this figure that sampling at the incorrect time causes the values at the output of the matched filter to be different from the values that should correspond to correctly timed samples. This error in sampling will cause the matched filter to no longer achieve zero ISI. This added ISI causes a great deal of degradation in the performance of the demodulator and must therefore be corrected.

In order to correct for this timing error a timing synchronization circuit is required in the receiver that can determine the correct sampling instant in order to properly reconstruct the transmitted data.

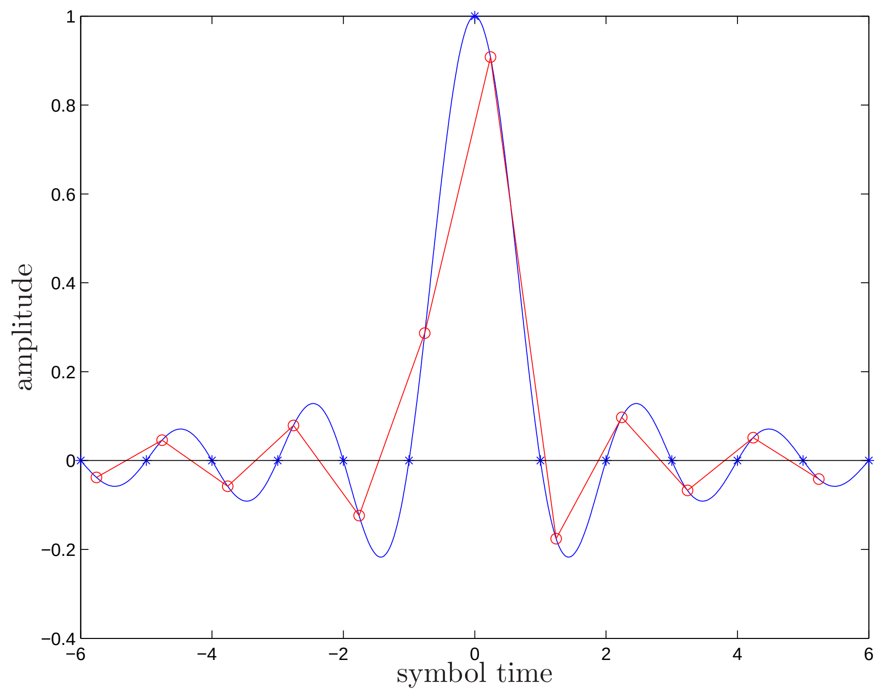


Figure 3.10: Effect of Timing Offset on Symbol Decision

Frequency Impairments

The hardware mismatch of the oscillators used in both the transmitter and receiver units, in addition to timing errors, can lead to an offset between the carrier frequency and phases in both units.

Frequency impairments occur when the demodulator and modulator do not have the exact same carrier frequencies. When this is the case the down-converted and filtered signals yield the terms in (3.6) and (3.7):

$$I_{rx}[n] = \frac{1}{2}(I[n]\cos((\omega_c - \omega_0)n) + Q[n]\sin((\omega_c - \omega_0)n) \tag{3.6}$$

$$Q_{rx}[n] = \frac{1}{2}(Q[n]\cos((\omega_c - \omega_0)n) - I[n]\sin((\omega_c - \omega_0)n) \tag{3.7}$$

where ω_0 and ω_c are the carrier frequencies of the modulator and demodulator, respectively. $I[n]$ and $Q[n]$ represent the in-phase and quadrature portions of the transmitted and received signal.

The resulting down-conversion causes the I and Q signals to no longer appear exactly at base-band, causing a frequency offset.

This offset causes the received I and Q constellation to appear to rotate. The constellation plot in figure 3.11 shows an example of a frequency mismatch of 0.0005 cycles/sample over 100 symbols. In the figure the red ‘x’ points indicate the ideal symbol locations and the blue circles represent the received values.

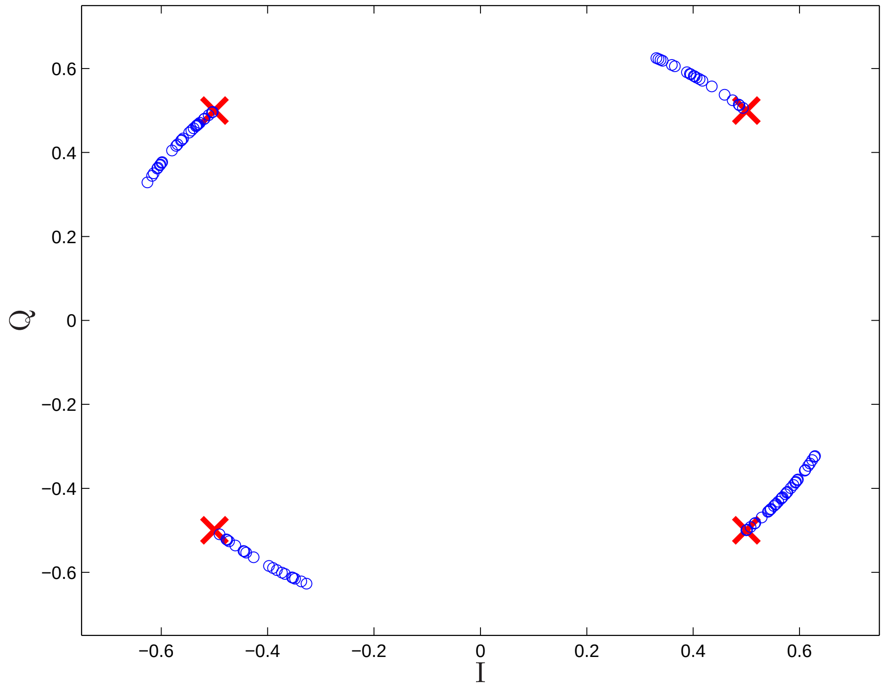


Figure 3.11: Effect of Frequency Offset on I and Q Constellation

This rotation causes the amplitude of the received I and Q symbol values to vary over time making a correct symbol decision impossible without correction.

Phase Impairments

In addition to the frequency impairments caused by mismatch between the oscillators in the transmitter and receiver, phase impairments may also occur. If the demodulator’s cosine and sine functions do not have the exact same phase as that of the modulator, the I and Q components of the received signals will appear to have a phase offset added to them. This phase offset can be expressed mathematically by adding a $\Delta\theta$ term to the sine and cosine terms in (2.12) and (2.13). After filtering the results are given by (3.8) and (3.9).

$$I_{rx}[n] = \frac{1}{2}\left(I[n]\cos((\omega_c - \omega_0)n + \Delta\theta) + Q[n]\sin((\omega_c - \omega_0)n + \Delta\theta)\right) \tag{3.8}$$

$$Q_{rx}[n] = \frac{1}{2}\left(Q[n]\cos((\omega_c - \omega_0)n + \Delta\theta) - I[n]\sin((\omega_c - \omega_0)n + \Delta\theta)\right) \tag{3.9}$$

This phase offset is described mathematically in (3.10) and (3.11) for both the received I and Q symbols respectively when no frequency offset is present.

$$I_{rx}[n] = \frac{1}{2}(I[n]\cos(\Delta\theta) + Q[n]\sin(\Delta\theta)) \tag{3.10}$$

$$Q_{rx}[n] = \frac{1}{2}(Qcos(\Delta\theta) - Isin(\Delta\theta)) \tag{3.11}$$

Without frequency offset, the phase offset then appears as a static rotation of the I and Q constellation as shown in figure 3.12. The red 'x' points indicate the ideal symbol placement and the blue circles represent the received values in the figure. This impairment will also result in an amplitude adjustment of the I and Q portions of the perceived received symbols making a correct symbol determination less likely.

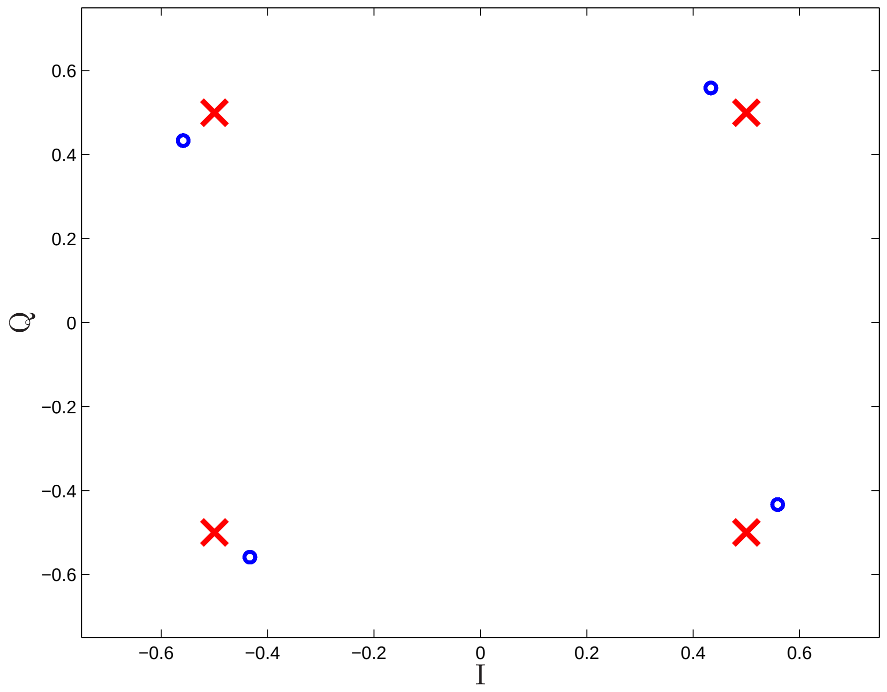


Figure 3.12: Effect of Phase Offset on I and Q Constellation

Noise Impairments

In addition to the other impairments previously mentioned, the received signal at the demodulator will also contain noise. Noise may be caused by thermal noise from the various electrical components that comprise the circuits for the transmitter and receiver as well as an additional circuitry that the signal has to pass through in the channel, such as amplifiers [27]. Noise is also caused from interference from outside sources. In a hybrid fiber/coaxial cable transmission system, although shielded, the cables can still pick up electrical noise from outside the cable. These outside sources can include lightning strikes, power transmission cables, switching power supplies and other natural or artificial sources [8].

In order to model this noise in a digital system, additive white Gaussian noise (AWGN) is generally used. The AWGN model is a good noise model since AWGN creates a zero-mean signal across all frequencies, creating a constant spectral density [27]. This type of signal can be used to represent the noise created by several different random processes as described by the central limit theorem [15].

The noise term, v_0 can simply be added to the end of (3.8) and (3.9) to form (3.12) and (3.13), which reasonably model the signals that may be received at the digital front end of the demodulator.

$$I_{rx}[n] = \frac{1}{2}\left(I[n]\cos((\omega_c - \omega_0)n + \Delta\theta) + Q[n]\sin((\omega_c - \omega_0)n + \Delta\theta)\right) + v_0[n] \tag{3.12}$$

$$Q_{rx}[n] = \frac{1}{2}\left(Qcos((\omega_c - \omega_0)n + \Delta\theta) - Isin((\omega_c - \omega_0)n + \Delta\theta)\right) + v_0[n] \tag{3.13}$$

Chapter 4

Hardware Constraints

4.1 Hardware Constraints

Prior to making design choices for the various circuits required for the implementation of an upstream DOCSIS based demodulator, some constraints are needed to be placed on the design. These constraints have been based upon the limitations of the hardware components being used for implementation and testing in addition to those detailed in the DOCSIS 3.0 physical layer documentation.

This chapter covers the constraints that have been imposed due to hardware selection for the example design presented and the reasons behind them.

4.1.1 FPGA selection

The FPGA development board used for implementation was chosen to be an Altera DE4 development board. The selection of the board was based on the board having a large number of logic elements and multipliers to provide room for development without initially needing to know the hardware costs of all the circuits that needed to be implemented.

The development board consists of a Stratix IV EP4SGX530KH40C2 FPGA and associated hardware required to interface with the FPGA. The EP4SGX530KH40C2 supports up to 424,960 combinational LUTs, 212,480 Memory LUTs, 424,960 logic registers and 512 18-bit by 18-bit multipliers.

4.1.2 Clock Frequencies

In the DOCSIS 3.0 all symbol rates are based upon a CM or CMTS having access to a 10.24 MHz local oscillator. The selected Altera DE4 development board has no such oscillator available that is any multiple of 10.24 MHz and the on-board PLLs available cannot accurately reproduce a 10.24 MHz clock signal.

The highest frequency oscillator directly available on the DE4 is 100 MHz. This oscillator was selected as the system clock for the design and implementation of circuits.

The DOCSIS 3.0 specifies the maximum symbol rate of upstream data transmission to be 5.12 Msym/s however the lack of a 10.24 MHz oscillator makes the implementation of this frequency impossible on the DE4 directly. Instead the symbol rate that has been implemented is 5 Msym/s, which is one twentieth of the highest frequency oscillator on the DE4 development board.Similarly, all clock frequencies within the design have been scaled from the DOCSIS to use the 100 MHz clock. Table 4.1 lists the clock frequencies that have been chosen for the circuits implemented throughout this thesis.

Table 4.1: Clock Frequencies Used During Implementation

Clock name	**Frequency (MHz)**
system clock	100
sample clock	20
symbol clock	5

Each of the clocks created are used for specific purposes in the design. The system clock is the highest rate clock to be used in the implementation of the front end demodulator. This high rate clock is used for the RF transmission sampling rate and high rate filtering. The sample clock is used to run the pulse-shaping and matched filters used for band limiting the digital pulses to be transmitted. This clock is run at four times the symbol clock rate in order to provide transmitted signals that have four samples for each transmitted symbol. The symbol clock is used to generate the symbols that are to be transmitted by the modulator and received by the demodulator.

4.1.3 Digital to Analog Converter

Throughout testing and implementation a digital to analog converter is used to verify frequency responses and signals from the implemented circuits.

The DOCSIS 3.0 specifies that the upstream frequency range of the CMTS must operate over a range of 5-85 MHz. The CM transmitting to the CMTS must only support a range from 4-42 MHz. The DAC used in this design is a Texas Instruments DAC5672 and supports sampling rates of up to 275 MHz. The DAC has been run at the system clock frequency of 100 MHz throughout this thesis in order to properly simulate the 4-42 MHz band that must be supported for data transmission.

4.1.4 Multipliers

The Stratix IV FPGA uses a number of DSP elements to implement individual multipliers. These DSP elements each contain four 18-by-18 bit multipliers that can be reconfigured into eight 9-by-9, six 12-by-12, four 18-by-18 or two 36-by-36 bit multipliers. These multipliers can then be configured into a number of different modes including individual multipliers, multiply and add or subtract and multiply and accumulate circuits. Many FPGAs do not contain DSP elements that can be configured into as many modes as those available on the Stratix IV and only have provisions for independent 18-by-18 bit multipliers. In order to allow for comparative purposes between FPGAs for the implementation costs of the circuits discussed throughout this thesis the DSP elements of the Stratix IV will be configured only as independent 18-by-18 bit multipliers.

Chapter 5

Performance Analysis and Design Choices

5.1 Performance Analysis

The DOCSIS 3.0 specifies the performance of many attributes of both transmitters and receivers based upon the modulation error ratio (MER) of the system. In order to analyze the quality of the demodulator front end being produced by following the methods presented in this thesis, the MER will be used as the primary means to determine the performance of the guided implementation presented here.

5.1.1 Modulation Error Ratio

In an ideal QAM system the received symbols map to a specific theoretical constellation point. In practical systems, however, the received symbol rarely falls exactly upon this theoretical constellation point. Sources of error, such as noise, spurious emissions, incorrect timing, frequency and phase offsets as well as other errors cause the received symbol to vary from the theoretical constellation point.

The MER is a measurement that computes the ratio between the average received signal power and the average power in the error of the received symbols. Figure 5.1 shows a visual representation of what the MER measures.

Considering that the MER takes into account all errors that affect the received symbol's location, it is an ideal measurement for the performance of the demodulator front end design presented in this thesis.

The MER, in dB, of a DOCSIS based receiver may be calculated using (5.1):

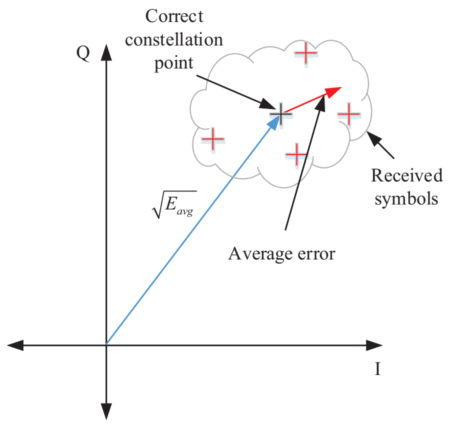


Figure 5.1: Modulation Error Ratio

$$MER_{dB} = 10\log_{10}\left(\frac{E_{avg}}{\frac{1}{N}\sum_{k=1}^{N} e_k^2}\right) \tag{5.1}$$

where E_{avg} is the average received symbol power, N is the number of symbols average over and e_k is the error vector from the correct constellation point.

5.2 Design of the Demodulator Front-End

QAM modulators and demodulators have been around for a number of years and are well known. The block diagrams shown in figures 5.2 and 5.3 show the blocks necessary to implement the DOCSIS based transmitter and front-end of the demodulator.

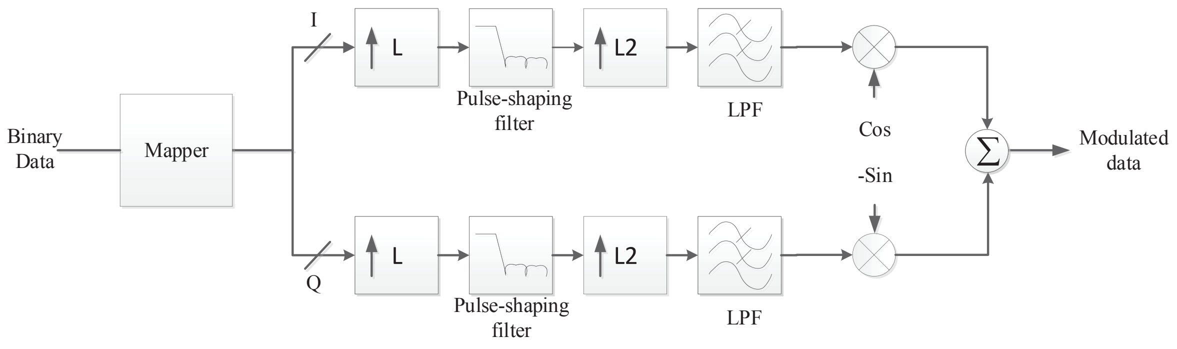


Figure 5.2: DOCSIS 3.0 Modulator Block Diagram.

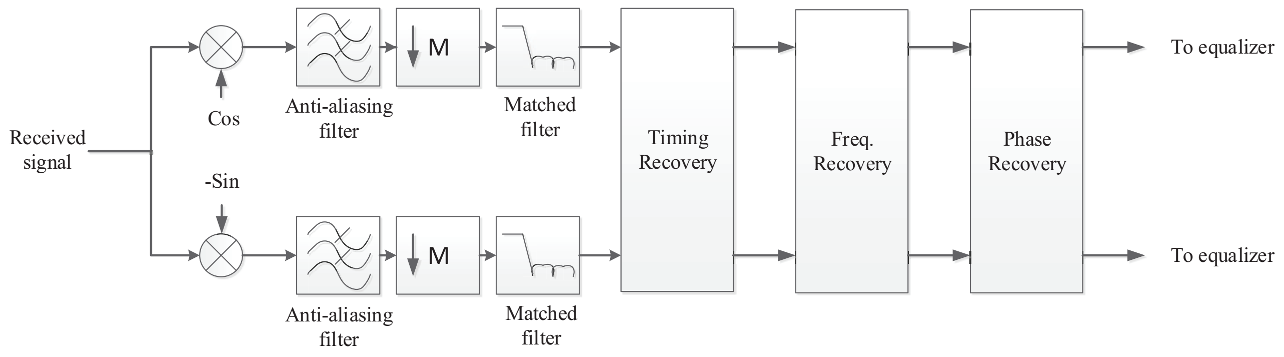


Figure 5.3: Block Diagram of Demodulator Front-end

In order to verify that the demodulator front end would function according to the DOCSIS with the design choices made, MATLAB was used to simulate the various blocks in the system and the receiver as a whole prior to implementation of the system in the DE4 FPGA.

This section will cover the theoretical breakdown of each block and the parameters used to design the various filters required for implementation.

5.2.1 The Pulse-shaping and Matched Filters

The DOCSIS is very clear about some of the design constraints that must be placed upon the modulator's pulse-shaping filter and the demodulator's matched filters.

In order to simulate and test a realistic demodulator the modulator must also be simulated. The first task is to then design a pulse-shaping filter that will behave according to the DOCSIS.

The DOCSIS states that the pulse-shaping filter must approximate a SRRC filter with a roll-off factor of 0.25 [7]. In addition, the DOCSIS lists a number of out-of-band emission requirements so that any transmitted signals from the modulator do not greatly interfere with any other channel transmissions on the same coaxial network.

The spectral out-of-band emission requirements for the pulse-shaping filter are listed in table 5.1. These specifications have been adjusted to use the clocking frequencies available based upon the hardware constraints as outlined in chapter 4.

For the matched filter, the specification calls for a SRRC that has a roll-off factor of 0.25. There are no such emission requirements for the matched filter as it is the job of the

Table 5.1: Out-of-band Emission Specifications for a 5 Msymbol Modulator

Frequency Range (Hz)	**Emission Power (dBc)**
6.25 MHz to 7 MHz	-58
7 MHz to 8 MHz	-60.5
8 MHz to 16 MHz	-63.5

modulator to reduce spectral emissions.

The DOCSIS uses MER to specify how well the modulator and demodulator must work together to properly receive a QAM signal. In the upstream channel the combination of the modulator and demodulator must achieve a MER of at least 30 dB.

Since the primary purpose of this thesis is to discuss the implementation and resource reduction of the demodulator the task at hand is to find a combination of pulse-shaping and matched filters that will yield a reasonable MER for the system while reducing resource usage. Since the matched filter should be a SRRC filter with a roll-off of 0.25, the only parameter that can be adjusted at this point for the matched filter is length.

In order to determine an optimum filter length for the matched filter simulations were conducted in MATLAB to vary the matched SRRC filter length. In all cases the SRRC filters were designed to use four samples per symbol. The matched filter was then cascaded with a pulse-shaping filter that had a large number of coefficients and met the out-of-band specifications as listed in table 5.1 and could achieve an MER of at least 50 dB.

The parameters chosen for the pulse-shaping filter chosen are shown in table 5.2.

Table 5.2: Parameters Chosen for Pulse-shaping Filter

Parameter	**Value**
length	32 symbols
r_{roll_off}	0.25
β_{kaiser}	0

The results of the MATLAB script are shown in figure 5.4.

To create a system that meets the provided specifications, a MER of greater than 30 dB must be chosen. While according to figure 5.4 a matched filter of 5 symbols in length could be chosen, a longer filter should be selected to give additional headroom from the effects of ISI caused by additional filtering and other impairments encountered throughout the upstream

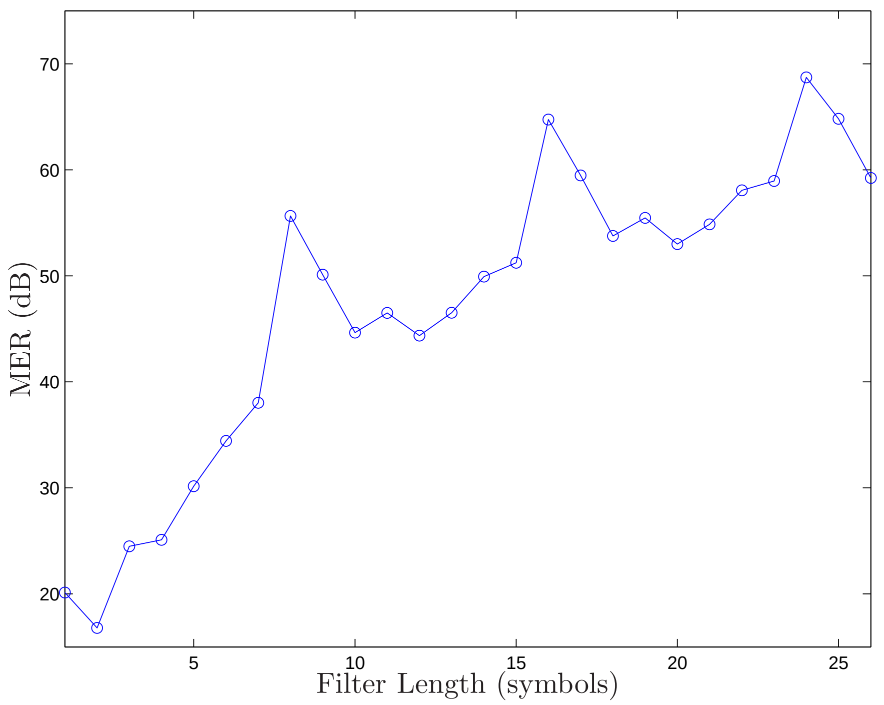


Figure 5.4: MER Vs. Matched Filter Length

channel. It makes sense to choose one of the local maxima points from the plot. A matched filter length of 8 symbols should provide a theoretical MER of 55.6 dB with no impairments, and this is the chosen length of filter that will be implemented.

5.2.2 Anti-imaging and Anti-aliasing Filters

From the hardware limitations discussed in chapter 4 the design will be limited to a 100 MHz sampling rate and have a symbol rate of 5 Msym/s.

The pulse-shaping and matched filters will be designed to be run at four samples per symbol meaning that the sampling rate of the filters must be 20 MHz. Upsampling the symbols by zero-stuffing causes images to be created in the modulator however the pulse-shaping filter removes these images. Further up-sampling by a factor of five is required to achieve the RF transmission sampling rate of 100 MHz.

This additional upsampling by five results in the images shown in figure 5.5 that must be removed so that they do not corrupt the transmitted signal or interfere with other channel transmissions. The filter that is used to remove the up-sampling images must have a stop-band frequency that starts at or before that given by (5.2).

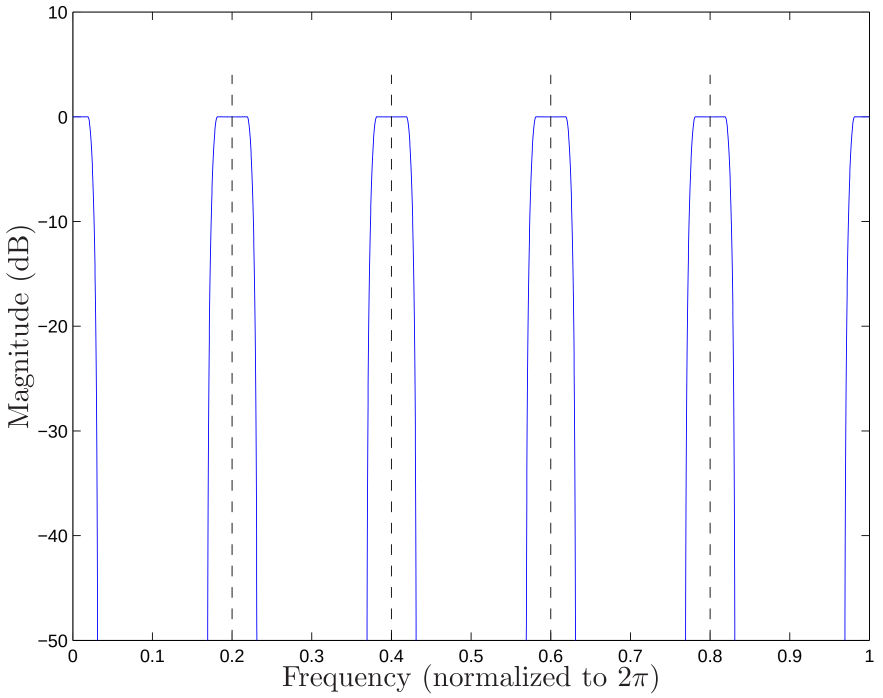


Figure 5.5: Spectrum of SRRC Upsampled by Five

$$f_{stop_{up}} = \frac{F_s}{L} - \frac{BW_{sig}}{2} \tag{5.2}$$

Where $f_{stop_{up}}$ is the stop band frequency, F_s is the sampling frequency of the anti-imaging filter. L is the up-sampling factor and BW_{sig} is the bandwidth of the transmitted signal.

Similarly the demodulator must downsample the downconverted RF transmission by a factor of five prior to entering the matched filter to achieve the proper sampling rate for the filter. This down-sampling process has the effect of expanding the bandwidth of the received signal relative to the sampling rate after down-sampling. If the bandwidth of the signal after down-sampling extends past half of the lower sampling frequency minus the bandwidth of the signal of interest corruption will occur in the down-sampled signal due to the overlap of the signal bandwidths. This type of corruption is referred to as aliasing.

The aliasing effect can be reduced by placing a filter with a stop band that occurs before or at the frequency point given by (5.3).

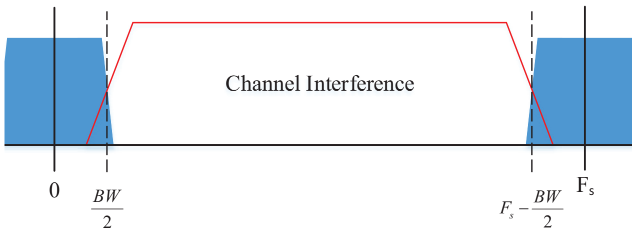
$$f_{stop_{down}} = \frac{F_s}{M} - \frac{BW_{sig}}{2} \tag{5.3}$$

Where f_{stop} is the stop band frequency and F_s is the sampling frequency of the anti-aliasing filter. M is the down-sampling factor and BW_{sig} is the bandwidth of the received signal.

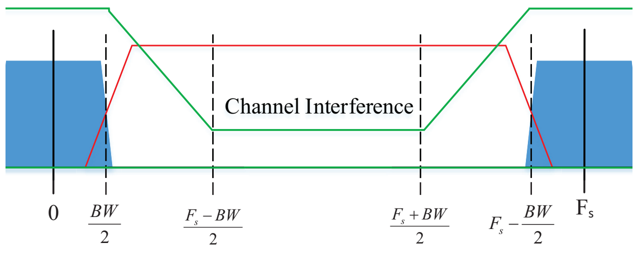
The figures 5.6a, 5.6b and 5.6c show the reason for needing the stop band to be located at the point given by (5.3). Figure 5.6a shows the received signal at RF sampling rates, including interference (noise and other transmissions) found in the upstream channel. Figure 5.6b shows the same received signal but overlays the required low-pass filter, shown in green, that will reduce aliasing effects. Figure 5.6c shows the downsampled received filter after filtering and shows how some energy from the channel interference, even after filtering, gets aliased back into the transmitted signal. The green and yellow lines in figure 5.6c show the low-pass filter after downsampling.

Considering the fact that both filters have the same stop-band corner frequencies the same filter design can be used for both anti-imaging and anti-aliasing.

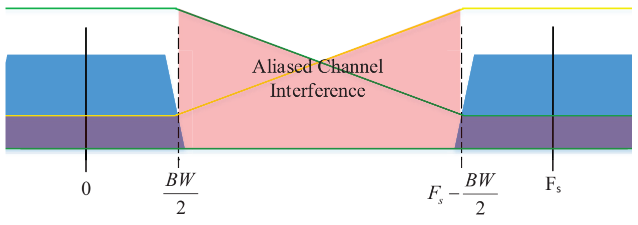
Given that the maximum theoretical MER of the system based upon the selection of the pulse-shaping and matched filters is about 56 dB the anti-aliasing and anti-imaging filters should be chosen to perform better than that specification. The filters examined in the following sections have been designed for at least 80 dB of attenuation, in order greatly reduce any images or spectral overlap from aliasing, and have a focus on reducing the number



(a) RF Channel with Interference



(b) Low-pass Filter to Remove Aliasing



(c) RF Spectrum after Filtering and Downsampling

Figure 5.6: Filtering to Reduce Aliasing Effects when Downsampling

of coefficients required to implement the filters.

In order to properly reconstruct the data that was transmitted to the demodulator the phase response of any filter used for anti-aliasing and anti-imaging purposes should have a linear phase response. If the filters do not have a linear phase response the time-domain signal containing the transmitted pulses may become distorted. This distortion may cause the decisions as to what symbol was transmitted to be incorrect and therefore result in incorrect received data.

Finite impulse response (FIR) filters typically have a structure as depicted in figure 5.7 containing a constant number of delays without any feedback paths. This constant number of delays provides a linear phase response at the output of the filter by delaying each time domain sample by the same amount.

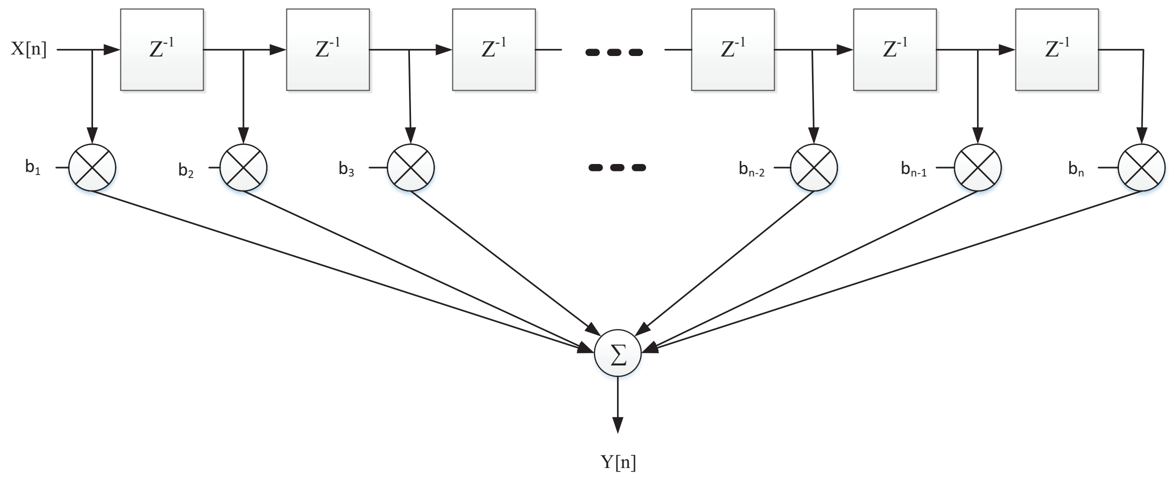


Figure 5.7: Typical FIR Filter Structure

In many communications systems half-band FIR filters are an ideal choice to produce both linear phase response and a reduction in coefficients. These half-band filters have the property that almost half of the coefficients in the filter are zero and therefore do not require multipliers. These filters work well for down-sampling by factors of two as their name suggests. The hardware constraints of this design example, however, do not allow for down-sampling by integer multiples of two but rather require a downsample by a factor of 5.

A non-half-band FIR filter that must meet the specifications as set forth in section 5.2.2 will require many multipliers depending upon the stop-band attenuation and pass-band ripple desired. A more multiplier efficient structure for these filters is to use near linear phase all-pass recursive filters.

Near Linear Phase All-pass Recursive Filters

All-pass recursive filters are a type of infinite impulse response (IIR) filter with the property that their magnitude response is unity. The fact that these filters are IIR allow them to have very sharp transition bands and achieve large attenuation with relatively few coefficients making them a good choice for anti-aliasing and anti-imaging.

The issue that usually occurs when using IIR filters is that the phase response of such filters are not linear due to feedback paths in the filter's structure. Fortunately these all-pass filters can be made to have a near linear phase response in their pass-band [14, 28]. Creating an all-pass filter network with the structure shown in figure 5.8 can achieve the near linear phase response desired when $H_0(z^2)$ is set to be a pure delay. The details of this are discussed by Harris [14] and will not be discussed further here.

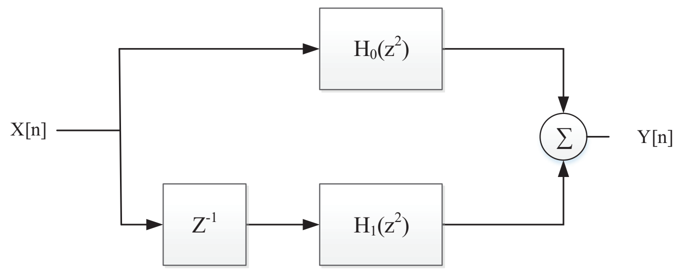


Figure 5.8: Two-path Near Linear Phase Filter Structure

All-pass filters can be used in M-path filter structures as shown in figure 5.9. Using the M-path structure results in the overall gain of the filter being M. It has been been shown that the pass-band 3 dB cutoff frequency of such a filter is given by (5.4) [28].

$$\omega_p = \frac{\pi}{M}\text{rad.} \tag{5.4}$$

Taking into account the transmitted signal bandwidth of 6.25 MHz that needs to be demodulated and using (5.4) it becomes clear that a 5-path filter will meet the pass-band requirements of the filters yielding a pass band corner of 10 MHz. While the passband corner

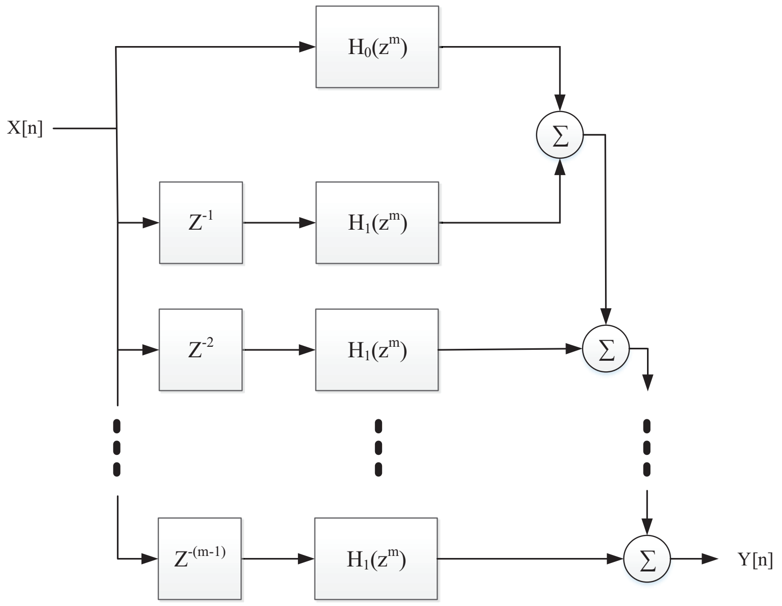


Figure 5.9: M-path Near Linear Phase Filter Structure

frequency will be met using an M-path near linear phase recursive all-pass filter, the stop band does not meet requirements.

The M-path near linear phase all-pass filters do not allow for zeros to be placed at frequencies of $\frac{\pi}{M} + \frac{2\pi}{M}k$ for integer values of k [28]. This causes the stop-band to have spurs at those frequency locations. Figure 5.10 shows the effect of not being able to place the zeros at the frequencies previously mentioned.

The spurs created by the M-path filters will have a bandwidth equal to twice that of the transition band between the pass-band cutoff frequency and the stop-band corner frequencies of the filter. This relationship is given in (5.5),

$$BW_{spur} = 2\left(\omega_s - \frac{\pi}{M}\right) \text{rad}, \tag{5.5}$$

where ω_s is the stop-band corner frequency and M is the number of filter paths being used.

In choosing the stop-band corner frequency for the 5-path filter it is necessary to understand that down-sampling by a factor of 5 will occur immediately after using the filter. This down-sampling will cause the transition band of the 5-path filter to alias back into the received the signal. To ensure that this aliasing does not adversely effect the received signal,

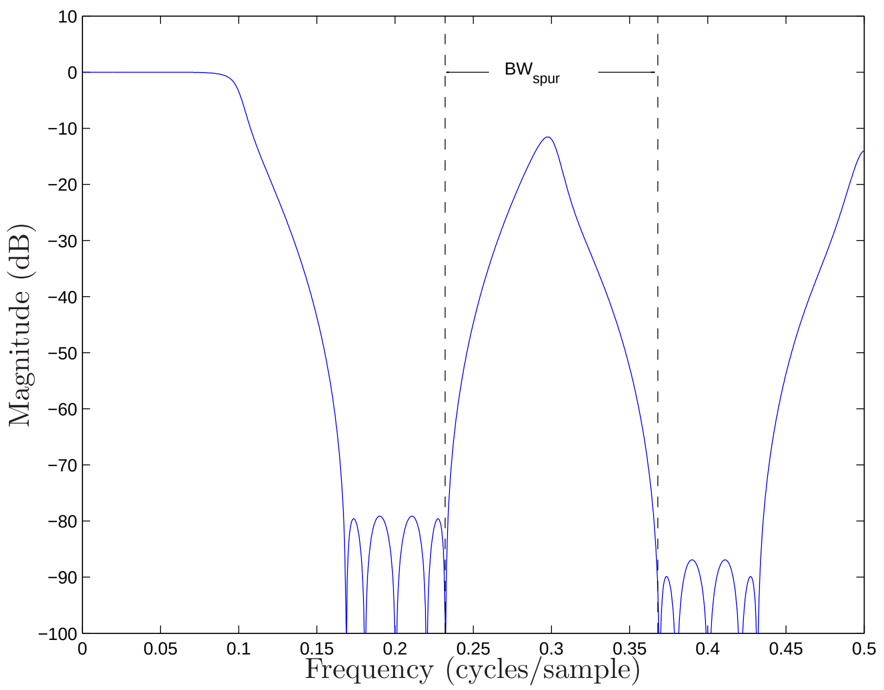


Figure 5.10: 5-path All-pass Filter Frequency Response

the stop-band corner of the filter must be chosen so that it meets the requirement given in (5.6).

$$\omega_s \leq \left(\frac{2\pi}{M} - \frac{BW_{sig}}{2}\right) \text{radians} \tag{5.6}$$

The choice of stop-band corner for this filter is dependent upon the attenuation that is desired. The higher the frequency of the stop-band corner the more attenuation that is achievable in the stop-band, this of course comes at the cost of widening the transition band of the filter. With the example design calling for a 6.25 MHz signal bandwidth, the stop-band corner must be located at $\leq$ 0.16875 cycles/sample or 16.875 Mhz.

In order to attenuate the spurs created by the 5-path filter, further filtering is required. A 3-path all-pass filter structure can be used to achieve this goal. The 3-path filter, however, has the same issue as the 5-path filter in that it produces a spur in the stop-band and it turn requires another filter. The three path filter frequency response can be seen in figure 5.11. For proper removal of the 5-path filter's spur the 3-path filter should have a stop-band corner that is located at a point that will achieve the desired attenuation of the spur created by the 3-path filter.

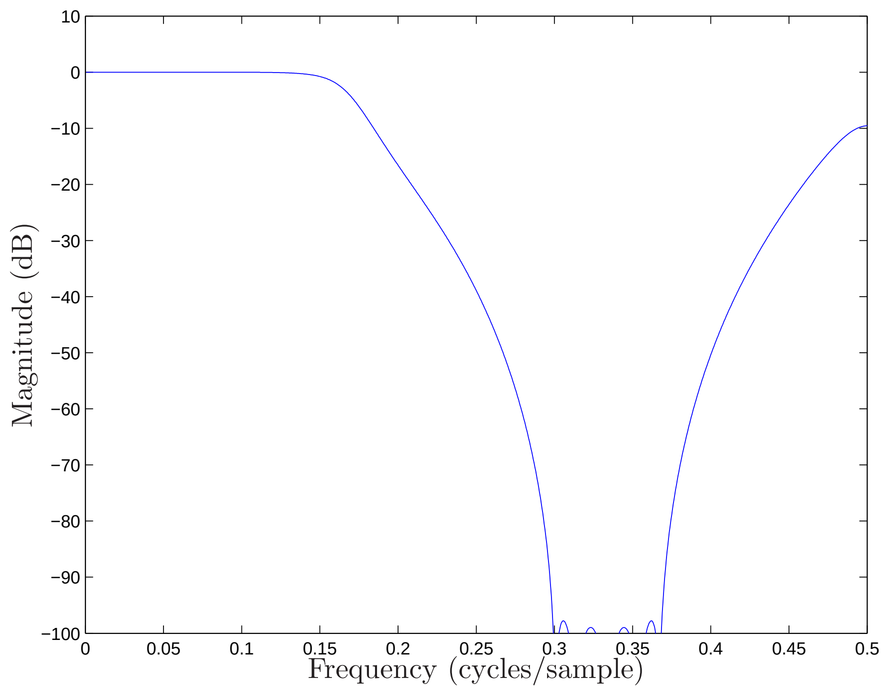


Figure 5.11: 3-path All-pass Filter Frequency Response

To attenuate the spur from the 3-path filter a 2-path filter can be used. The 2-path filter does not suffer from the spurs present in the 3-path or 5-path filters. Again it is important to choose the stop-band corner of the 2-path filter so that it will achieve the desired attenuation of the spur.

The two path filter and cascaded filter frequency responses can be seen in figures 5.12 and 5.13, respectively. Figure 5.13 shows the pass-band corner location at 0.1 cycles/sample, the stop-band corner at 0.16875 cycles/sample, the attenuation at the stop-band corner of 80 dB with black dashed lines, and the point that aliasing will come back to after down-sampling occurs with a red dashed line.

The resulting filters produce a near linear phase response in the pass-band of the filter as shown in figure 5.14. The linear-phase is most important over the transmitted signal bandwidth at baseband which contains the transmitted data. The required bandwidth at baseband is half of the total signal bandwidth required during RF transmission. Figure 5.15 and 5.16 show the phase response and error in the phase response compared to a perfectly linear phase filter over the required signal bandwidth at baseband, respectively.

Although it may seem like it would be more efficient to create an M-path FIR filter to

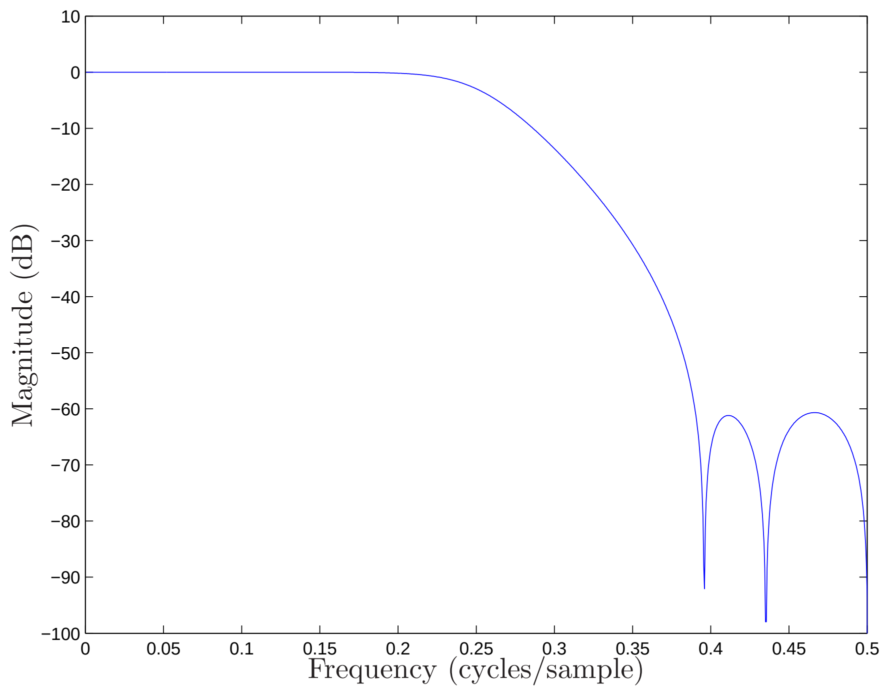


Figure 5.12: 2-path All-pass Filter Frequency Response

preform the anti-imaging and anti-aliasing required, the number of coefficients in the all-pass structure is still less than that of an FIR filter that meets the same stop-band attenuation. In order to design the anti-aliasing filter's coefficients an open-source MATLAB script written by Dragan Vultec and Fred Harris was used [13]. The tables 5.3, 5.4 and 5.5 provide the coefficients that were used to implement the 2-path, 3-path and 5-path stages of the anti-aliasing filter used in this example design, respectively. The M-path all-pass filter design presented here achieves a stop-band attenuation of better than 80 dB and requires a total of 14 coefficients.

For comparative purposes, figure 5.17 shows the magnitude frequency response of the all-pass filter design and that of a equiripple FIR filter and a cascaded-integrator-comb (CIC) filter. The equiripple FIR based filter, with pass-band corner of 0.1 cycles/sample, a stop-band corner of 0.168 cycles/sample and an attenuation of 80 dB with 0.01 dB ripple, would require about 49 coefficients. The CIC filter examined here is described by Rice and uses no multipliers, however it requires an FIR filter to compensate for droop across its passband [27]. The addition of an equiripple FIR compensation filter to the CIC structure, in order to achieve a similar specification to that of the FIR filter previously mentioned, requires 100

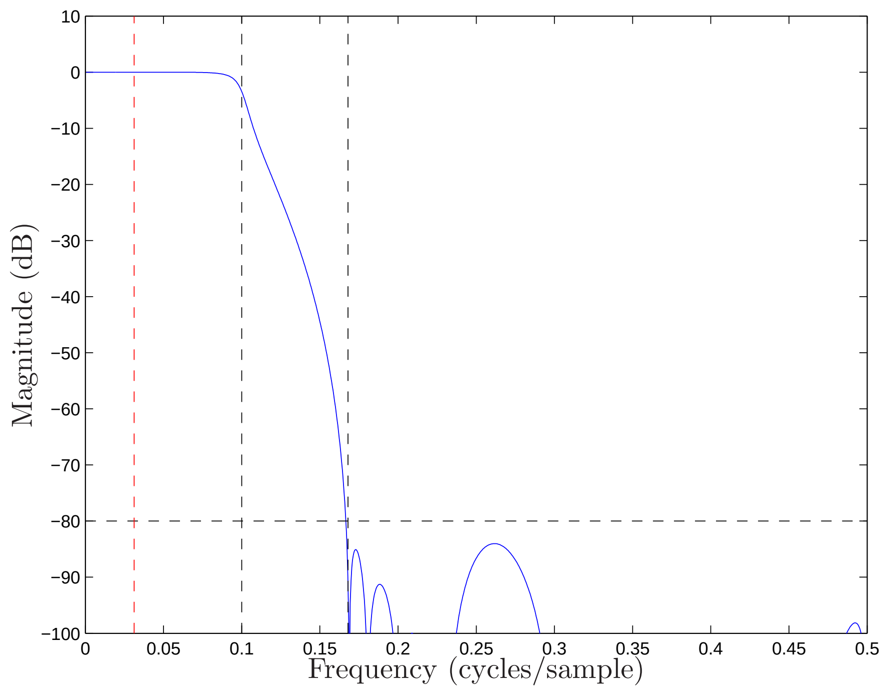


Figure 5.13: Cascaded M-path All-pass Filter Frequency Response

coefficients to achieve a 0.01 dB passband ripple over DC to 0.05 cycles/sample.

Since the all-pass based filter uses fewer coefficients than the other methods examined, the example design carried out here will make use of the designed all-pass filter structure.

Table 5.3: 2-Path Allpass Filter Coefficients for Stop-band Corner at 0.395 cycles/sample, with 4 delays in first path

path	**coefficient 1**	**coefficient 2**
2	0.506580364780383	-0.08218670440503492

Table 5.4: 3-Path Allpass Filter Coefficients for Stop-band Corner at 0.3 cycles/sample, with 6 delays in first path

path	**coefficient 1**	**coefficient 2**
2	0.3298654678968272	-0.07537841054588207
3	0.6256039264903217	-0.04907021304683482

Table 5.5: 5-Path Allpass Filter Coefficients for Stop-band Corner at 0.168 cycles/sample, with 10 delays in first path

path	**coefficient 1**	**coefficient 2**
2	0.2338543048779194	-0.08295025989259303
3	0.4007854558910866	-0.08018149458071466
4	0.5760758821438221	-0.06217925246160427
5	0.7721556235589868	-0.03483985415107635

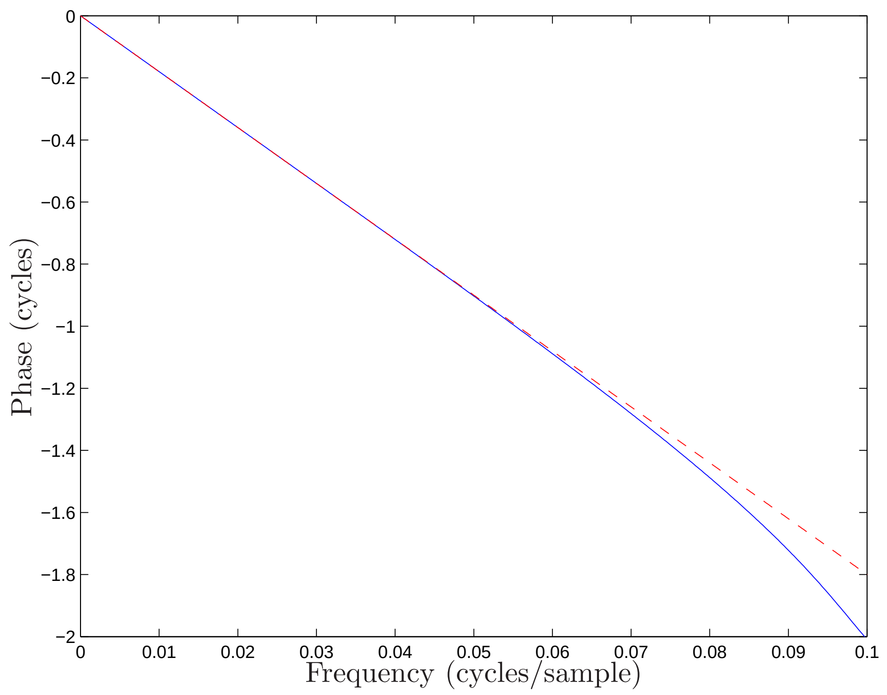


Figure 5.14: M-path All-pass Filter Phase Response

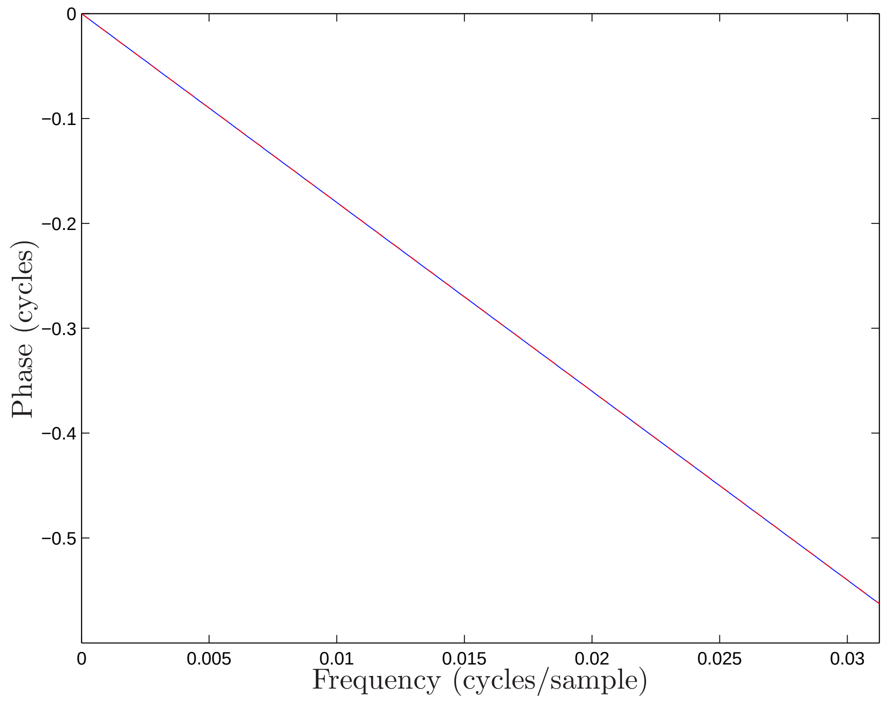


Figure 5.15: Cascaded M-path All-pass Filter Phase Response

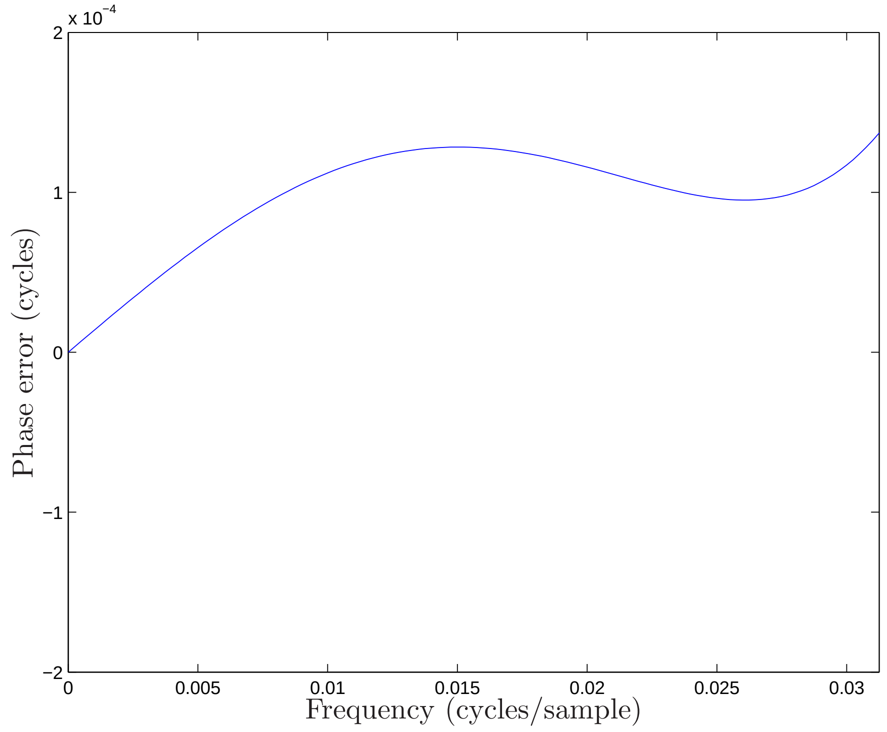


Figure 5.16: Cascaded M-path All-pass Filter Phase Error

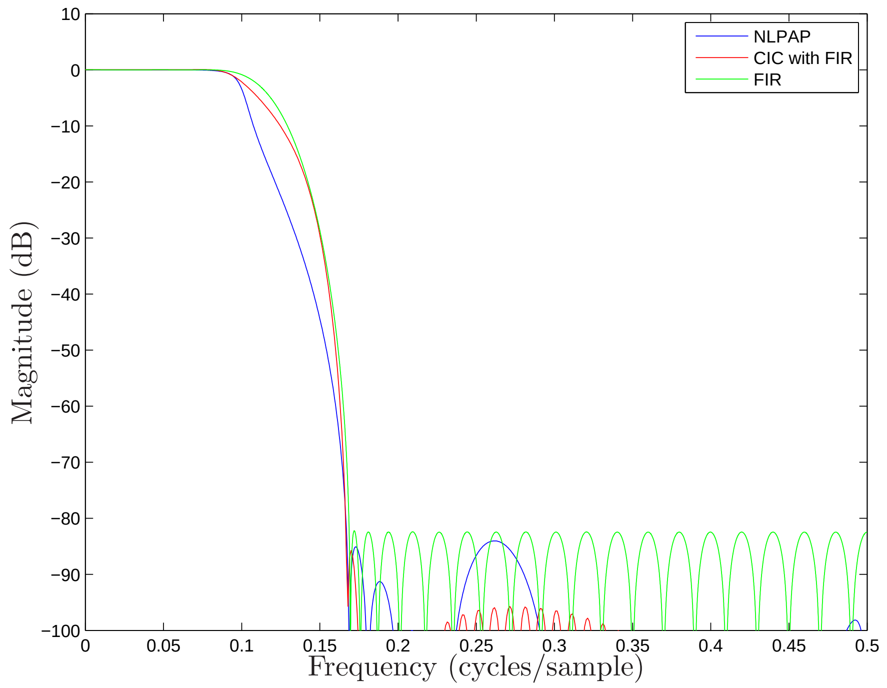


Figure 5.17: Magnitude Response Comparison of FIR, CIC and Near Linear Phase All-pass (NLPAP) Filters

5.2.3 Feedback and Feedforward Synchronization

There are numerous techniques that can be used to achieve proper synchronization between the transmitter and receiver circuits in a QAM based communication system. A number of these techniques use feedback systems that tend to require a relatively large amount of time to converge upon the proper compensation value. The DOCSIS upstream requires data to be sent in bursts of various lengths requiring that synchronization be completed in a short amount of time. These short synchronization requirements have lead to the use in this thesis of feedforward compensation techniques for synchronization.

5.2.4 Timing Recovery

The first stage in synchronization to recover the transmitted symbols at the receiver is to sample the incoming signal at the proper time. Any sampling time offset between the modulator and demodulator when making a decision as to what data were transmitted causes a great deal of error. The figure in 5.18 shows the output of the matched filter, for one of the QAM phases, at the demodulator with respect to the sampling timing used at the demodulator to determine the transmitted symbol. It becomes clear from this plot that as the sampling time moves farther away from the correct sampling time of a transmitted symbol that the amount of error becomes increased, making a correct decision almost impossible, as any of the blue points could be chosen at any given sampling time as the transmitted symbol.

Any timing offset has an overall degradation on the MER of the system. The plot in figure 5.19 show the amount of achievable MER for the system with respect to incorrect sampling time.

Although many timing recovery techniques exist, such as the one presented by Gardner in [11], the convergence time of the feedback loop used in this recovery circuit is not well suited for TDMA burst mode communications systems. Instead of the use of feedback timing synchronization, feedforward timing synchronization is used to reduce the synchronization convergence time.

The block diagram in figure 5.20 shows the basic structure of the feedforward timing synchronization circuit that is used here and is based upon that presented in [17]. Immediately

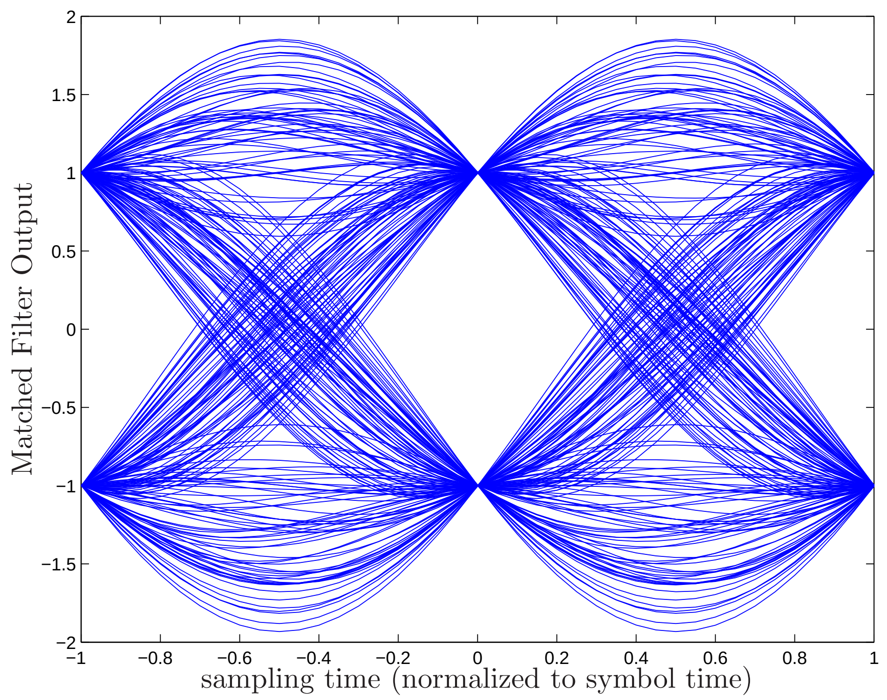


Figure 5.18: Eye-diagram of Timing Offset

after the anti-aliasing filter and first downsampler in the demodulator the matched filter is then used to remove any remaining downconverted interference from the received signal, I_{dn} and Q_{dn}. The filtering results in only the original transmitted channel and some residual down-conversion aliasing and attenuated interference remaining.

In a four sample per symbol system, the matched filter is run at four times the symbol rate of the system resulting in the best sampling time that can be recovered at the output of the matched filter to be one quarter of the symbol rate.

One quarter of the symbol rate is not a very fine timing adjustment to perform timing recovery since the RF channel was run at a rate of five times that. In order to achieve a better resolution for the timing synchronization circuit the output of the matched filter should be upsampled back to the RF transmission rate. Once upsampled the signal must once again be filtered to remove images caused by the upsampling. This filtering can be performed with the anti-aliasing or anti-imaging filters previously designed or a different filter can be used as long as it meets the stop band requirements given by (5.6). The filtering has the effect of interpolating between sample points at the output of the upsampler flowing the matched filter. It is then possible to achieve timing accurate to one-twentieth the symbol rate of the

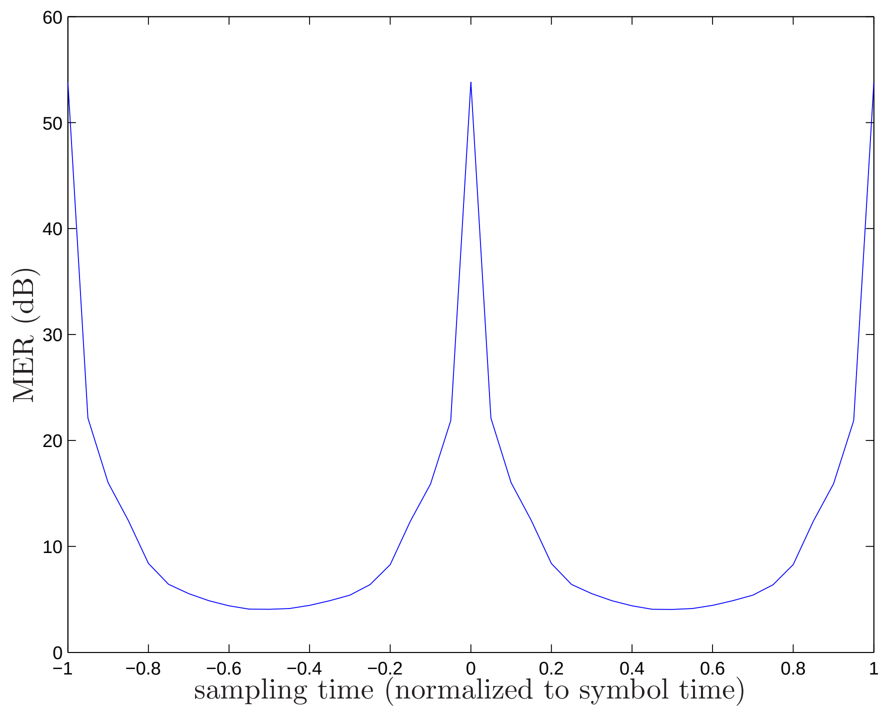


Figure 5.19: MER Vs. Timing Offset

system designed here.

In order to use a feedforward compensation technique a unique preamble packet must be sent. The DOCSIS states that the preamble must be a quadrature phase-shift keyed (QPSK) signal, equivalent to that of a 4-QAM signal. The preamble to be used is up to the designer of the system, however certain preambles may produce better results than others.

The preamble signal chosen for use in the design presented here is a 13 symbol long Barker code sequence. The Barker sequence was chosen for its auto-correlation properties. This type of sequence is relatively short and has an auto-correlation as shown in figure 5.21. Since there are only two possible data points on each the I and Q phases in a QPSK preamble, the Barker auto-correlation yields a distinct peak when the correct sequence is received that can be used to identify the correct timing of symbols.

When using the auto-correlation of a preamble to determine the proper timing synchronization for the demodulator the potential phase and frequency offsets that can occur must also be taken into account. If the phase or frequency offsets cause the symbols to be out by 180 degrees the auto-correlation peak will be inverted. The sum of the output power of the auto-correlation filters used in each phase in the demodulator can be used to counter these

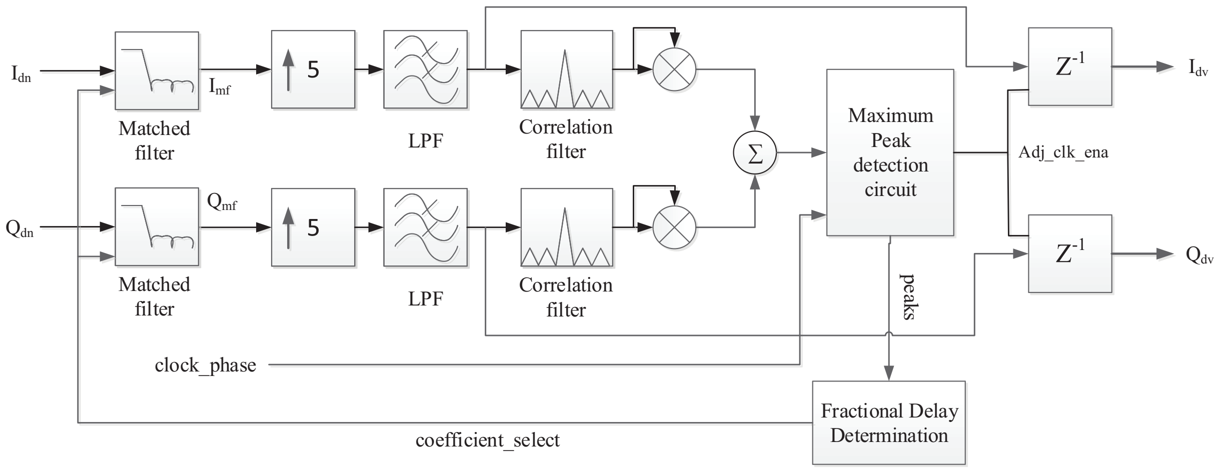


Figure 5.20: Timing Recovery Block Diagram

offset effects.

The squared outputs of the correlation filters are passed into a maximum peak detection circuit. This circuit as its name suggests searches the incoming samples to find the largest peak present in its input over a period of one preamble packet. The circuit keeps track of the relative sampling time at which the maximum peak occurs to that of the system clock, given by the "clock_phase" input signal. The clock_phase input signal is simply a counter that keeps track of number of system clock samples that occur between the symbol clock samples, in this case from 0 to 19.

After achieving timing accuracy to one-twentieth the symbol rate it is still possible to further refine the timing adjustment. It is possible to change the matched filter coefficients to create timing adjustments that are fractions of a sample. The filter coefficients can be evaluated using equation (3.1) at times equal to a fraction of a sample. These pre-evaluated coefficients can then be stored in memory and accessed to create a fractional delay through the SRRC filter.

To select the appropriate set of coefficients to implement a fractional delay within the matched filter, a fractional delay determination circuit is used. This circuit takes the maximum peak and adjacent samples to that peak determined by the maximum peak detection circuit as an input and calculates the fractional delay to be used once the maximum peak has been detected. Using these samples the fractional delay determination circuit can use equation (5.7) to fit the peak and adjacent samples to a parabola and determine any residual

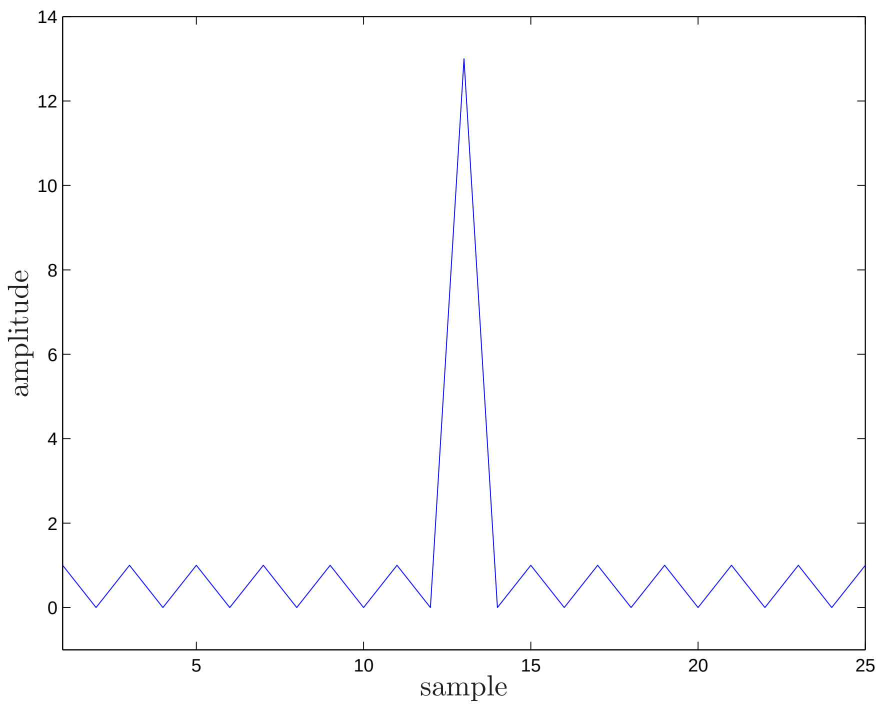


Figure 5.21: Auto-correlation of Length 13 Barker Sequence

timing offset [17],

$$\tau = \frac{|p[n-1]| - |p[n+1]|}{2(|p[n+1]| + |p[n-1]| - 2|p[n]|)}, \tag{5.7}$$

where $p[n-1]$, $p[n]$ and $p[n+1]$ are the samples previous to the maximum peak, at the maximum peak and just after the maximum peak as determined by the maximum peak detection circuit, respectively.

Once the auto-correlation peak is located the demodulator is then set to sample the output of the interpolator at that time and every twenty system clock samples thereafter. After this re-sampling, the output rate of the timing synchronization circuit will be equal to that of the symbol rate of the system.

After simulation of the timing recovery circuit, the MER of the system was determined to be 54.7 dB, a degradation of 0.9 dB from the theoretical response of the cascaded pulse-shaping and matched filters.

5.2.5 Frequency Recovery

Traditional frequency recovery circuits make use of Costas loops [27]. These loops can be adapted from the traditional analog domain to the digital domain as presented in [9]. While these techniques work very well for continuous signals they tend to be less optimum for short burst communications such as those found in DOCSIS 3.0 compliant networks. Instead of using a Costas loop, a feedforward technique that makes use of a preamble is used here. Since a QPSK preamble is already chosen to be used for the timing recovery circuit it makes sense to also use this preamble for the frequency recovery circuit, and in doing so allows for the use of a feedforward structure.

The feedforward frequency recovery circuit that operates over three preamble packets examined in this thesis has been previously proposed and well studied by Berscheid [6, 5]. The block diagram of the proposed frequency recovery circuit is shown in figure 5.22.

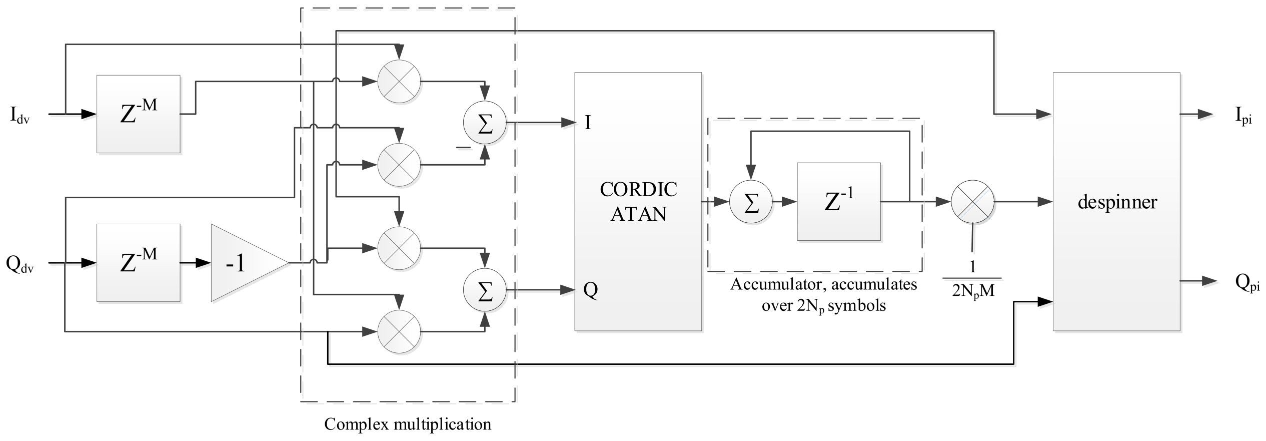


Figure 5.22: Frequency Recovery Block Diagram

In order to determine the frequency offset the circuit first captures an entire preamble sequence and stores the sequence in a series of registers, one register for each symbol of the preamble. This series of registers are referred to as a delay chain and is shown in figure 5.22 as Z^{-M}. After the registers are filled, the next preamble packet begins to enter the synchronization circuit. The current preamble value and the complex conjugate of the stored symbol corresponding to the current symbol are multiplied together. The results of this multiplication are given in (5.8),

$$\begin{aligned} S_{detector} &= A_{pre}e^{j\Delta\omega n}e^{jM\Delta\omega n}A_{pre}e^{-j\Delta\omega n} \\ &= A_{pre}e^{j\Delta\omega n(1+M)}A_{pre}e^{-j\Delta\omega n} \\ &= A_{pre}^2 e^{jM\Delta\omega n}, \end{aligned} \tag{5.8}$$

where A_{pre} is the preamble symbol being examined, $\Delta\omega$ is the frequency offset after down-conversion, and M is the number of delays in the delay chain in figure 5.22.

Taking the angle of (5.8) yields $M\Delta\omega$ for the multiplication of the current preamble symbol and the delayed preamble symbol. The angle is determined by taking the arctan of the I and Q outputs of the complex multiplication in figure 5.22. The arctan results are then accumulated and averaged over three preamble periods to determine the frequency offset as described in (5.9),

$$\omega_{\text{offset}} = \frac{1}{2N_p(M)} \sum_{n=0}^{2N_p-1} M\Delta\omega_n, \tag{5.9}$$

where N_p is the length of the preamble sequence and $M\Delta\omega_n$ is the output of the arctan function at sample n and M is the length of the delay chain.

The frequency offset found in (5.9) can then be directly used to correct the offset frequency by use of a despinner. The despinner effectively performs a complex multiplication of the input signal by $e^{j\omega_{\text{offset}}}$ the result of which is given in (5.10).

$$\begin{aligned} S_{despin} &= S_{in}e^{-j\Delta\omega n}e^{j\omega_{\text{offset}}n} \\ &= S_{in} \end{aligned} \tag{5.10}$$

Where S_{in} is the complex input signal to the despinner circuit.

Since the function of the complex multiplication and arctan block in figure 5.22 is to compute the angle between the delayed preamble symbols and current preamble symbols at each symbol instant, the same result can be achieved by using two arctan functions and taking their difference.

The reason for changing from the complex multiplication into two separate arctan circuits is to reduce the number of multiplications throughout the frequency recovery process and this being a suitable place to do so. It is proposed to use the frequency synchronization circuit shown in the block diagram in figure 5.23. The CORDIC based circuits used in this solution are discussed in detail in Chapter 5. The use of the CORDIC blocks results in the removal of all multipliers that are required to implement the arctan and complex multiplication in figure 5.22.

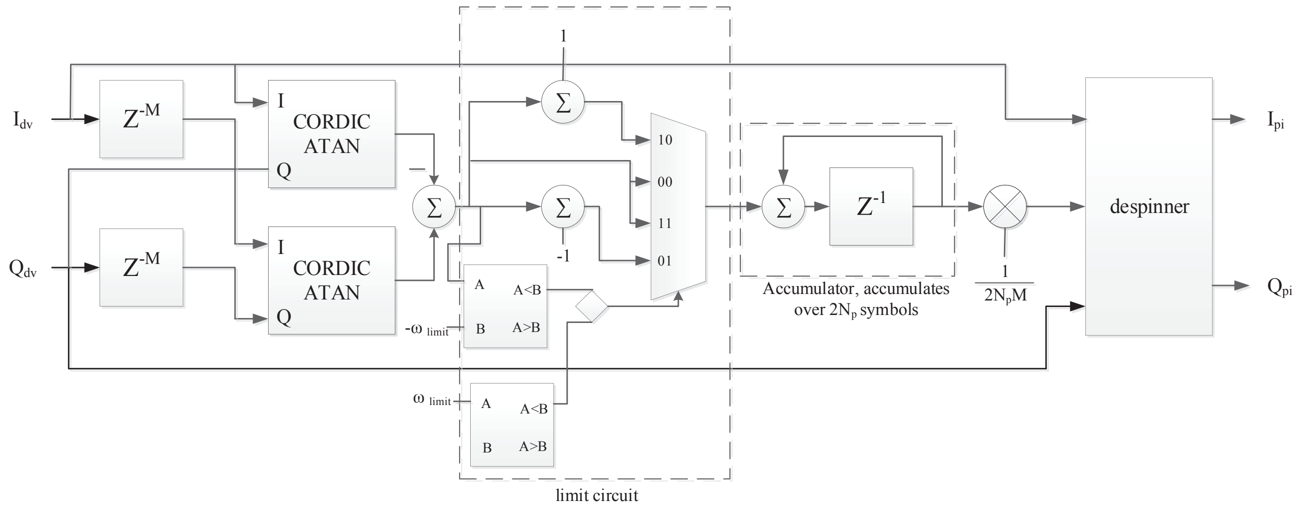


Figure 5.23: Frequency Recovery Block Diagram with CORDIC Blocks

Care must be taken when using the arctan blocks as proposed. Since the arctan function can be adapted to give results over $[-\pi, \pi]$ cases can occur where the difference between the two arctan blocks can yield a phase inversion (i.e. exceed $\pm\pi$).

This phase inversion will cause the averaged value of the frequency offset to no longer converge to the correct point. Fortunately some limits can be defined to overcome this phase inversion problem.

The DOCSIS defines that the maximum frequency drift between the receiver and transmitter oscillators is to be 50 parts per million. In this system that results in a maximum frequency deviation of 5 kHz between the transmitter and receiver that must be corrected for. This means that the amount of change between the delayed and current preamble symbols in the synchronization circuitry should not vary by more than ±0.013 cycles/sample as given by (5.11),

$$\Delta f_{\text{offset}} = \pm 50\text{x}10^{-6}\left(M\frac{F_{sys}}{F_{sync}}\right), \tag{5.11}$$

Where M is the preamble delay in the synchronization circuitry, F_{sync} is the sampling frequency of the synchronization circuit and F_{sys} is the frequency of the system's oscillator.

The limits of the frequency synchronization circuit can be set using (5.12) to achieve the required Δf_{offset},

$$\omega_{limit} > \Delta f_{\text{offset}}, \tag{5.12}$$

where Δf_{offset} is the range over which the frequency synchronizer needs to converge in cycles/sample.

From (5.12) and from (5.11) the frequency limit for the synchronization circuit should be greater than 0.013 cycles/sample.

To counter the possible phase inversion the difference between the two arctan blocks can be examined to see if they result in a value greater than ω_{limit} cycles/sample or less than $-\omega_{limit}$ cycles/sample. If the values are greater than ω_{limit} then 1 cycle/sample can be subtracted from the result. If the values are less than $-\omega_{limit}$ then 1 cycle/sample can be added to the result. This correction removes the phase inversion effect. Care should be taken when selecting the limit values to use to counter the phase inversion effects of the two arctan blocks. If any additional noise sources are present in the preamble symbols they will affect the limits that should be chosen.

Noise sources can cause the preamble symbols to have larger or smaller peak values than initially expected. If the magnitude of the symbols is larger than expected, the limit circuit used to prevent phase inversion may incorrectly invert the phase of a received signal if the noise pushes it past the limit thresholds set by (5.12). The limit values should therefore be adjusted to compensate for any reasonable amount of noise that could be expected in the system.

Since the frequency offset range that is expected in the DOCSIS upstream channel is very small relative to the maximum sampling rate of the system, the limits of the phase inversion correction circuity can be set quite high, up to 0.5 cycles/sample, to accommodate a large amount of noise in the preamble symbols. Using ω_{limit} equal to 0.5 cycles/sample yields a synchronization circuit that could achieve synchronization with frequency offsets of up to 0.0385 cycles/sample without noise interference and as such has been the value selected for implementation.

The difference of the arctan functions can then be accumulated and averaged according to (5.9) to determine the frequency offset to be used for synchronization purposes.

The plots in figures 5.24 and 5.25 show the simulated results of the synchronization range of the frequency synchronization circuit when ω_{limit} is set to 0.5 cycles/sample without noise and with 25 dBc AWGN, respectively. The vertical black dashed lines indicate the theoretical values of ω_{limit} and the vertical green lines indicate the required frequency offset, Δf_{offset}, range as specified in the DOCSIS. The blue dashed line indicates the actual frequency offset of the incoming signal and the red line indicates the frequency offset value determined by the proposed synchronization circuit.

Simulations were conducted on the frequency recovery circuit using 10000 packets of three 13 symbol long Barker sequences to determine the average error and variance of the circuit's simulation.

Figure 5.26 shows a plot of the average frequency offset difference between the proposed circuit and the actual frequency offset of the incoming symbols for both no noise and 25 dB AWGN. The variance of the simulated circuit is also given over a range of SNRs in figure 5.27. The variance of the arctan implemented circuit agrees with the theoretical variance given by (5.13) [6],

$$\sigma^2 = \frac{1}{M^2(N-M)SNR}\text{rad}^2, \tag{5.13}$$

where M is the number of preamble delays at the input to the frequency estimation circuit and N is the number of preamble symbols to average over.

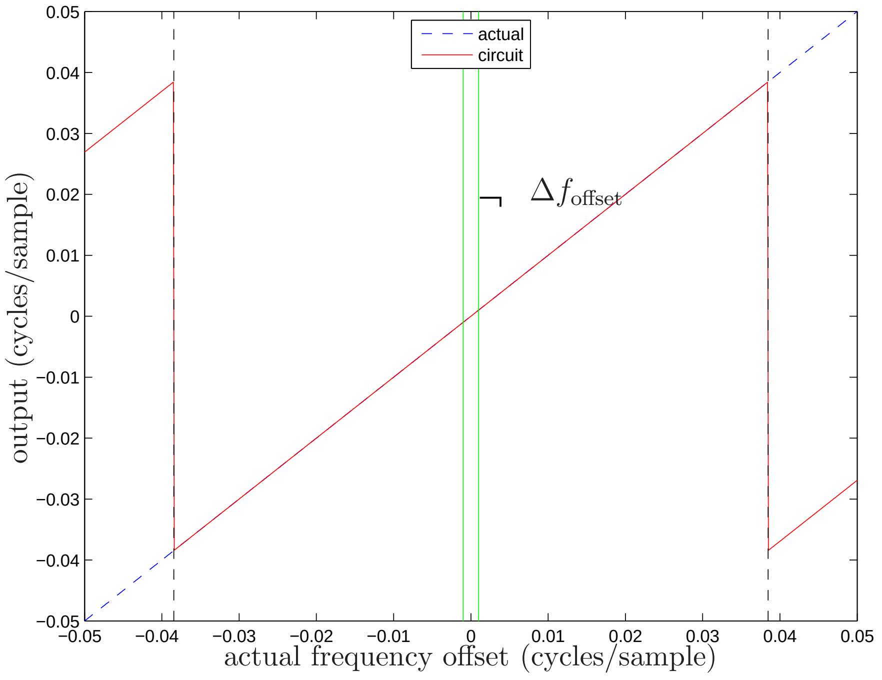


Figure 5.24: Limits of Frequency Synchronization Circuit with $\Delta f_{limit} = 0.5$ cycles/sample

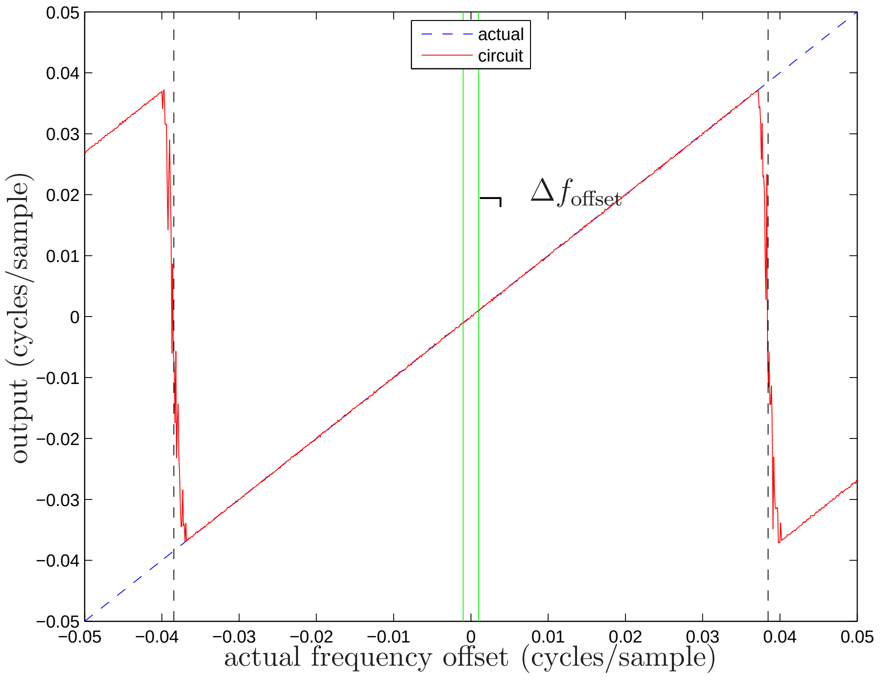


Figure 5.25: Limits of Frequency Synchronization Circuit with $\Delta f_{limit} = 0.5$ cycles/sample and 25 dB AWGN

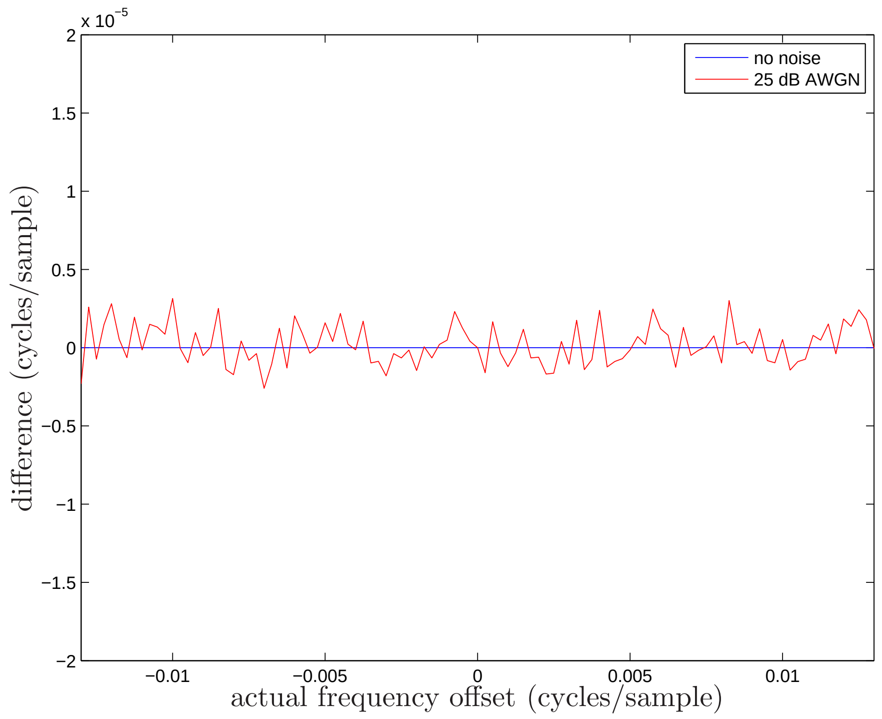


Figure 5.26: Frequency Synchronization Circuit Error with $\Delta f_{limit} = 0.5$ cycles/sample

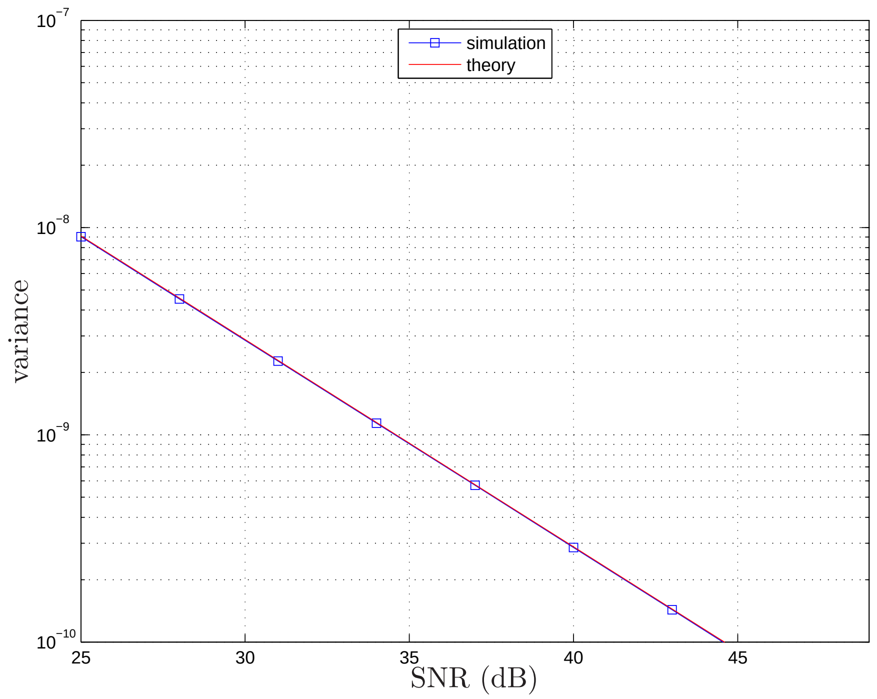


Figure 5.27: Frequency Synchronization Circuit Variance

5.2.6 Phase Recovery

A feedforward phase recovery circuit has been previously presented by Berscheid and is shown in the block diagram in figure 5.28 [6].

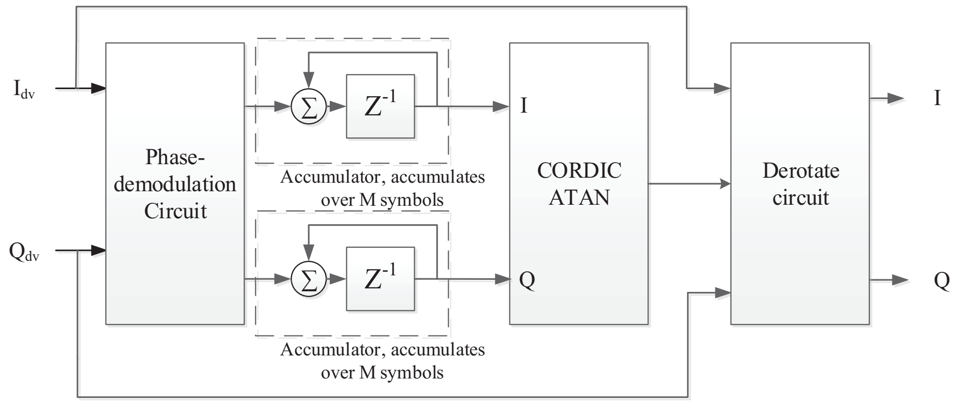

I_{dv}
Q_{dv}
Phase-demodulation Circuit
Σ
Z^{-1}
Accumulator, accumulates over M symbols
Σ
Z^{-1}
Accumulator, accumulates over M symbols
I
Q
CORDIC ATAN
Derotate circuit
I
Q

Figure 5.28: Phase Recovery Block Diagram

The phase recovery circuit relies on advanced knowledge of what the QPSK preamble signal is. The preamble signal as mentioned earlier has been chosen to be the following length 13 Barker sequence: 1111100110101. The Barker code is then converted to a QPSK signal and transmitted. Each of the zeros and ones in the Barker sequence are converted to the QPSK constellation points shown in figure 5.29. The mapping of the Barker sequence to the QPSK constellation has the effect of creating a phase modulated signal with each symbol of the preamble being either 0 or 180 degrees apart from one another.

Once the frequency synchronization has been achieved, the input to the phase synchronization circuit is given by (5.14),

$$S_{pi} = A_{pre} e^{j(\theta_n + \theta_{\text{offset}})}, \tag{5.14}$$

where A_{pre} is the amplitude of the preamble symbol, θ_n is the phase of the preamble symbol, and θ_{offset} is the phase offset that the synchronization circuit must correct for.

The first step in the phase recovery is then to remove the phase modulation on the input signal created by the QPSK coding of the Barker sequence. Since the preamble sequence being

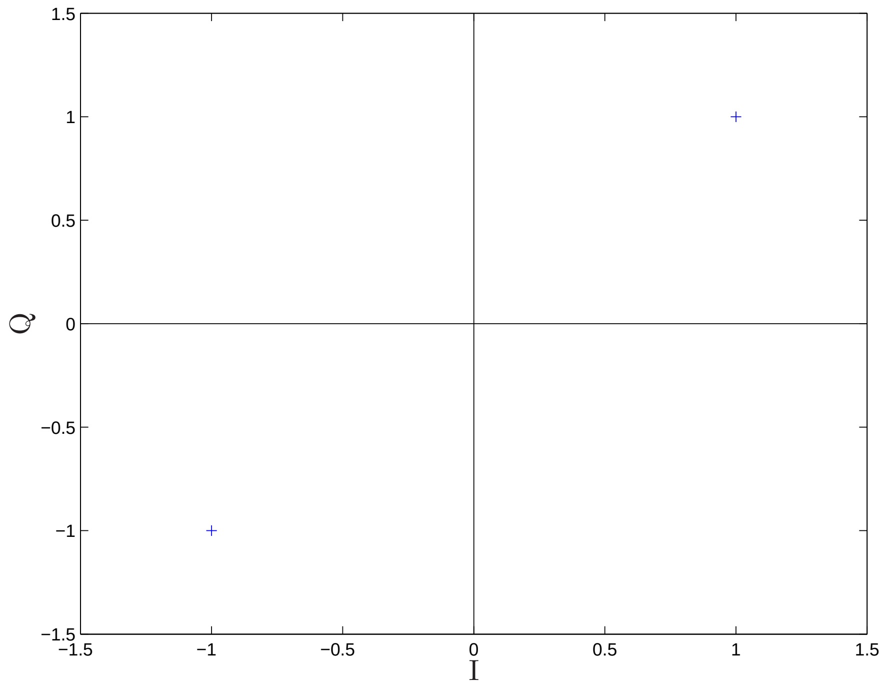


Figure 5.29: QPSK Constellation Diagram

used is known ahead of time, the phase demodulation can be accomplished by multiplying the phase recovery circuit input signal by the complex conjugate of the QPSK preamble signal as shown in (5.15).

$$\begin{aligned} S_{pd} &= A_{pre} e^{-j\theta_n} S_{pi} \\ &= A_{pre}^2 e^{j\theta_{\text{offset}}} \end{aligned} \tag{5.15}$$

After phase demodulation has been accomplished the phase synchronization can be completed by accumulating the complex conjugate of the I and Q signals over one or more preamble sequences and taking the arctan of that accumulation. No averaging of the accumulated values is required since the accumulation of the I and Q signals is a vector addition and inherently produces a phase average. Berscheid has previously shown that the variance of such a phase synchronization circuit is given by [6],

$$\sigma^2 = \frac{1}{2N\text{SNR}}, \tag{5.16}$$

where the SNR is the signal to noise ratio of the incoming signal in linear terms and N is the number of preamble symbols begin averaged over.

The computation of the phase difference between the I and Q signals can be achieved by taking the arctan of the phase demodulated signal S_{pd} and can be implemented with a CORDIC based approximation circuit.

The phase correction is then performed upon the input signal to the phase synchronization circuit via a complex multiplication by $e^{-j\omega_{\text{offset}}}$. In doing this multiplication it is once again convenient to use a CORDIC based circuit to remove the multipliers.

In order to verify that the circuit would behave as expected, a MATLAB simulation was done to compute the variances at different SNR values on a single preamble packet and compare these to those presented by Berscheid. SNRs ranging from 40 dB to 25 dB in steps of 3 dB were created by adding AWGN into a transmitted preamble signal with a phase offset ranging between -0.35 and 0.35 cycles in steps of 1/100 of a cycle. For each of the phase offsets added to the noise corrupted preamble packet, the simulation was run 500 times and variances were then calculated. The results of the simulation are given in figure 5.30 along with the theoretical results. The simulation indicated that the circuit, with multiplier reductions, should perform as well as the one proposed in [6].

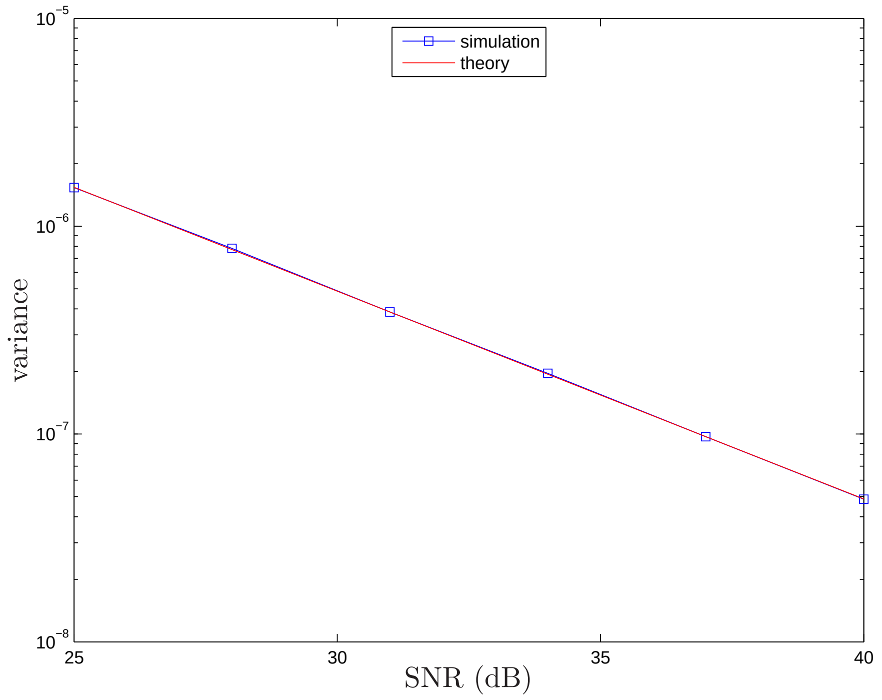


Figure 5.30: Phase Variance Vs. SNR

A second simulation was conducted to see the effect of varying the number of symbols used to calculate the phase offset in the recovery circuit. The simulation was performed over the same phase offset range in the previous simulation, however the SNR was fixed at 25 dB, the worst case SNR as defined in the DOCSIS. For each phase offset the simulation was run 2000 times and a variance calculated. The simulation was then rerun with a different number of symbols used to calculate the phase offset. The results of the simulation are given in figure 5.31 along with the theoretical values. It can be seen that increasing the number of symbols used in the synchronization circuit causes the overall variance to improve however small adjustments in the number of symbols used do not greatly effect the variance as expected. Since increasing the number of symbols used does not greatly affect the variance a reasonable design choice for the number symbols is 13, the same length as a preamble packet.

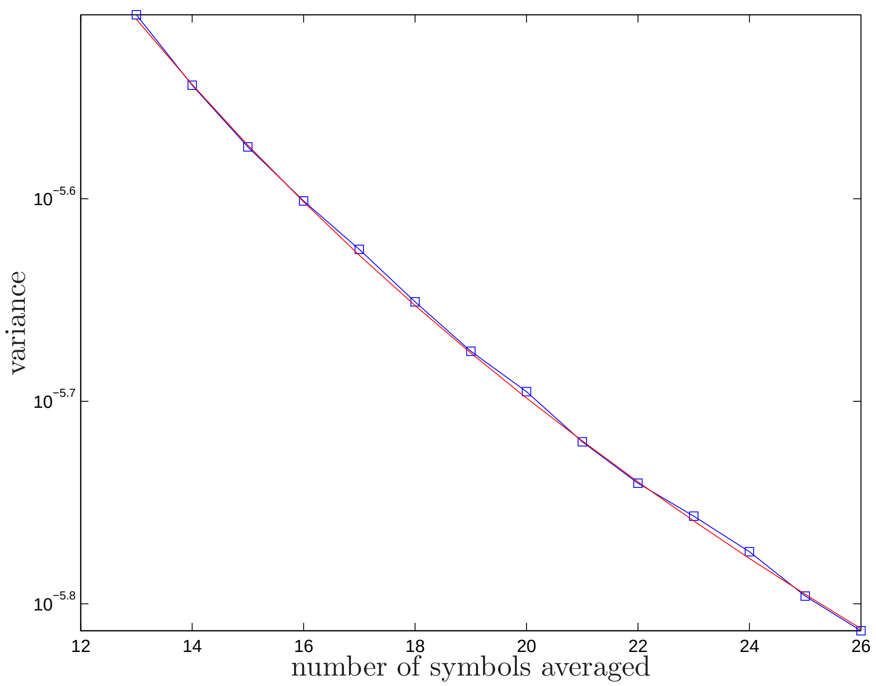


Figure 5.31: Phase Variance Vs. Symbols Used

Chapter 6
Digital Front End

6.1 Digital Front End

This chapter discusses the practical implementation of the circuits required to implement the digital front-end of a DOCSIS 3.0 based upstream receiver. The focus areas of the chapter will be bit sizing of the hardware structures, implementation structures, resource reduction techniques, and performance and testing of the circuits designed based upon the methods discussed in chapter 5.

6.2 Circuit Conventions

In order to better understand the circuits presented throughout this chapter it is beneficial to review the circuit schematic symbols and signal format conventions used. These symbols and conventions are discussed in the following sections.

6.2.1 Digital Number Formats

Since practical circuits can only consist of a finite number of bits, any number to be represented by a binary signal must be quantized. Each binary bit can either consist of a 1 or 0 giving each bit two possible states. For each bit that is added to a binary signal the maximum integer number that can be represented by that signal increases by a factor of two. The maximum integer value that can be represented by an unsigned binary signal can be mathematically given by equation (6.1),

$$A = 2^N - 1, \tag{6.1}$$

where N is the number of binary bits used to represent a binary signal, and where N must be greater than zero.

For the case of signed binary values the bit with the greatest value, known as the most significant bit (MSB), is used to represent the sign bit of the signal, making the maximum negative value that can be represented by the binary signal to be -2^{N-1}. Each bit that follows the sign bit has its value added to the sign bit to determine the overall signal value. As an example a binary signal consisting of 5 bits could represent integer values of [0,31] or [-16,15] when treated as unsigned and signed respectively.

These binary signals can also be represented as fractional numbers, both signed and unsigned. These signal formats are used throughout this thesis by using the XuY and XsY notations for unsigned and signed fractional values respectively. The X value represents the number of integer bits used in the signal format and Y to represent the fractional bits used for the signal. The fractional value of such a signal can be determined by treating the signed or unsigned value as an integer value of X+Y bits and dividing it by the integer value of the fractional bits as given in (6.2),

$$A = \left\lfloor \frac{A}{2^{N_f}} \right\rfloor, \tag{6.2}$$

where A is the value of the binary signal and N_f is the number of fractional bits used to represent the signal. An example using the binary value 10111 to represent a 1u4 and a 1s4 signal yields $\lfloor \frac{23}{16} \rfloor = 1.4375$ and $\lfloor \frac{-9}{16} \rfloor = -0.5625$, respectively.

6.2.2 Circuit Symbols

Various symbols are used throughout the circuits presented in this chapter. In order to ease the readability of the schematics presented, these symbols are discussed here.

As previously mentioned the binary signals used throughout the implemented circuits can take on unsigned or signed formats in both integer or fractional formats. The symbols in figure 6.1 show how the formats will be represented on the schematics. Wires may contain a single slash with the number of bits, N, presented above, below or beside the wire containing the slash. These wires indicate integer formatted signals that may be either signed or unsigned. Notation beside wires using the XuY format indicate unsigned fractional values, while XsY notation indicates signed fractional values.

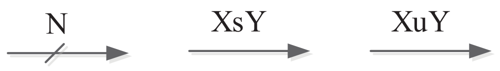


Figure 6.1: Signal Format Symbols

Registers used in the schematics presented are denoted by a box containing a Z^{-N}, where N represents the number of registers chained together in series. All sequential and register circuits presented in the schematics that follow are assumed to be run with the system clock feeding their clock inputs. Sequential logic and register blocks containing a `E:X` value indicate that the blocks use one or more enable signals. The enables run to each block will be noted at the top left hand corner of the schematics presented. The X value(s) following the `E:` indicate the enable signal that should be connected to the enable input of the block and will correspond to the `E:X` signals listed in the top left corner of the schematics.

Both combinational and sequential logic circuits may have multiple inputs and outputs. A small triangle is placed in the corner of sequential logic blocks to indicate the need for a clock signal to drive the logic contained within.

The blocks used to represent the registers, combinational and sequential logic circuits are shown in figure 6.2.

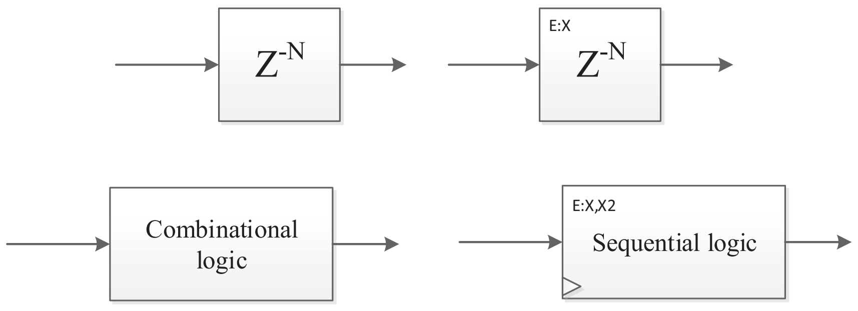


Figure 6.2: Register, Combinational and Sequential Circuit Symbols

There are instances where the number of bits used to represent signal formats must be changed to implement specific circuits. These operations are represented by the symbols in figure 6.3. Signals can be sign extended by replicating the most significant bit N_a times, or have N_c most significant bits removed. Signals can also have an additional N_d least significant bits added to them, or N_b least significant bits removed from them.

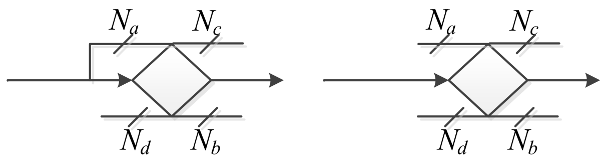


Figure 6.3: Signal Truncation and Extension Symbols

In cases where a specific bit must be extracted from a group of bits in a signal (known as a bus), the symbol in figure 6.4 is used. The value N is used to indicate the specific bit in the input signal S_{bus}, made up of N_s bits, that is to be extracted from the input signal. For example, if S_{bus} is a signal made up of 18 bits, numbered 1 through 18 from least significant to most significant, and N is 1, the first bit would be extracted from the signal.

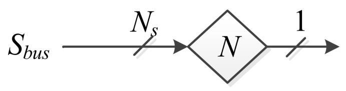


Figure 6.4: Bit Extraction Symbol

The circuit symbol shown in figure 6.5 refers to a multiplexer. This circuit can accept a number of inputs labeled from 0 to N in the figure. The function of the multiplexer is similar to that of a switch or valve. The select signal indicates which input is to be connected to the output, sig_out, of the multiplexer. The select signal of course must take on a value from 0 to N in order to accomplish this task.

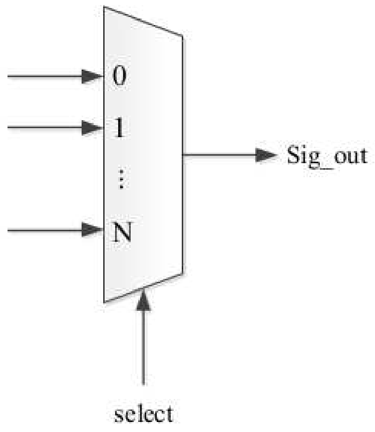


Figure 6.5: Multiplexer Symbol

6.3 Pipelining

During practical implementation of digital circuits it becomes important to ensure that the setup, hold and propagation times required for combinational logic circuits can be met. When combinational logic circuits such as adders and multipliers are used in cascade the propagation time from the input of the circuit to the end result of the circuit becomes increased. This increased time from the input of the circuit to its output sets the maximum frequency that the circuit can be run at in order to ensure that a valid output arrives at or before the next sampling time. This becomes especially important in signal processing applications where samples are expected to be taken at a constant rate.

In order to ensure that propagation times in combinational circuits are small enough to achieve the desired sampling rate of the circuits, additional registers (delays) can be inserted between combinational elements to effectively reset the propagation time at the input to the next combinational circuit at the expense of the overall circuit taking an extra sample delay time to complete.

The concept of inserting delays to increase the overall clock rate of a circuit is illustrated in figure 6.6. In the top portion of figure 6.6, two combinational circuits have been placed in cascade. The propagation times through the circuits are t_{p1} and t_{p2}, respectively. The total time for a valid input signal to propagate through the circuit becomes the sum of the two

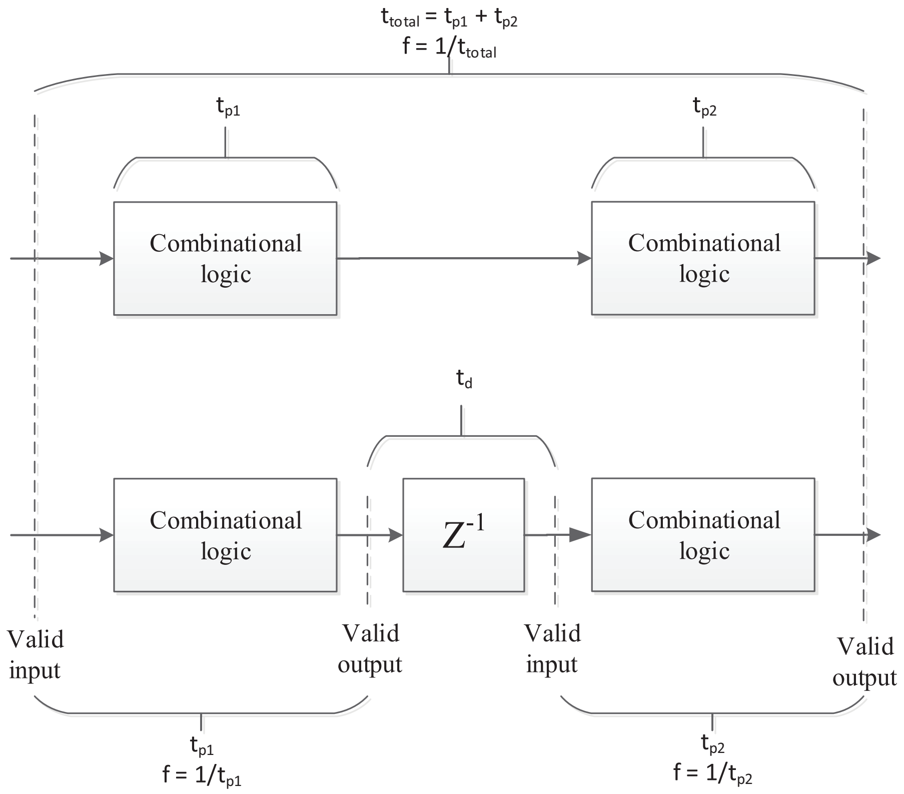


Figure 6.6: Pipelining of Circuits

propagation times, t_{total}. The total time then sets the maximum frequency that the overall circuit can be run at to ensure a valid output is achieved. An input or output is considered to be valid if the signal at the input or output of the combinational circuit has the correct output at the required sampling time.

The bottom portion of figure 6.6 illustrates how adding a pipelining delay between the two combinational circuit can increase the maximum frequency that the overall circuit can be run at. In this case the total time from the input of the first combinational circuits to the output of the final combinational circuit increases by the delay time amount t_d. The delay causes the output of the delay to become a valid input to the next combinational circuit. By adding this pipelining delay the maximum frequency that the overall circuit can be sampled at becomes the minimum of the inverse of the individual propagation times of the combinational circuits used.

This pipelining technique is used throughout the circuits presented in this thesis in order to achieve the required sampling rates of the circuits.

6.4 Down-conversion

The first stage in the digital front end is to downconvert the transmitted signal back to base-band.

In an FPGA the sine and cosine signals required for the down-conversion process can be implemented using an numerically controlled oscillator (NCO). An NCO can be constructed with a read-only memory (ROM) and an accumulator as shown in figure 6.7. The function of the ROM is to hold the sample values that constitute a single cycle of a sinusoidal signal. The accumulator takes the phase increment value (frequency, f_{in}) and adds it to the accumulator's output. This output in turn gives the address of the sample stored in the ROM that corresponds to the phase of the sinusoid stored in the ROM. As the accumulator continues to add values to its output, the phase of the sinusoid produced at the output of the ROM continues to increase at each sample instant, thereby producing a sinusoidal signal.

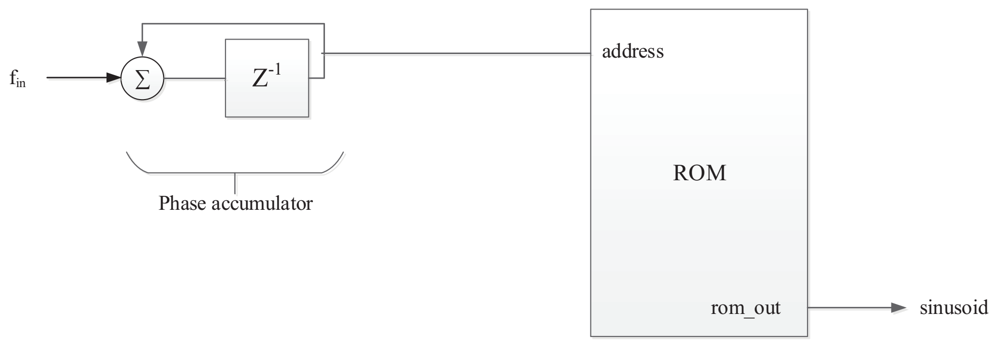


Figure 6.7: Block Digram of ROM based NCO

A cosine can be produced from the same ROM if a two port ROM is used and the address to the second input port of the ROM is offset by adding one quarter cycle from the first input port.

The sine and cosine outputs of the NCO can then be multiplied by the received signal to down-convert the received signal to base-band. This down-conversion scheme is somewhat costly from a hardware point of view as two multipliers running at the sampling rate of the input to the receiver must be used to perform the down-conversion process.

A more cost effective solution for the down-conversion process that does not use any multipliers is to use a CORDIC based downconverter.

6.4.1 CORDIC Downconverter

The CORDIC demodulator is based upon the algorithm presented by Volder in 1951 [32]. The algorithm uses a successive approximation technique to compute the sine and cosine of a given input vector and phase angle.

The technique works by passing the input vector's x and y components through several iterations. Each iteration is performed by examining the input phase angle to the sum of a set of either the negative or positive of predefined phase increment values for each iteration up to the present iteration. If the input phase angle to each iteration is less than the current iteration's sum of phase increments then the current iteration's phase increment value is subtracted from the sum; otherwise it is added to the sum. The current iteration's sum is then presented as the phase input to the next iteration. The comparison between the iteration's input phase angle and the iteration's phase increment value determines whether a portion of the x coordinate is added or subtracted from the y coordinate and likewise, whether or not a portion of the y coordinate is added or subtracted from the x coordinate. This process is then repeated until the number of iterations specified are completed. It follows that the more iterations that are performed the more accurate the results become.

The output of each CORDIC iteration yields (6.3),

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} cos(\theta_k) & -sin(\theta_k) \\ sin(\theta_k) & cos(\theta_k) \end{bmatrix} \begin{bmatrix} x_k \\ y_k \end{bmatrix}, \tag{6.3}$$

where k is the iteration number and θ_k is the phase angle increment for that iteration.

Factoring out the cosine term from (6.3) and replacing the factored cosine term with (6.4) yields (6.5).

$$cos(\theta) = \frac{1}{\sqrt{1 + tan^2(\theta)}} \tag{6.4}$$

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \frac{1}{\sqrt{1+tan^2(\theta_k)}} \begin{bmatrix} 1 & -tan(\theta_k) \\ tan(\theta_k) & 1 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \end{bmatrix} \tag{6.5}$$

The multipliers required to multiply x_k and y_k by the $tan(\theta_k)$ can be removed by selecting the values of θ_k to be $\frac{\pi}{2}$ radians for the first iteration and $atan(2^{2-k})$ for each additional iteration. The selection of these values allows the use of bit shifts to perform the multiplications.

It is clear that (6.5) has a gain associated with it and the correction of this gain would require a multiplier. This gain is dependent upon the number of iterations used to implement the CORDIC circuit and is given by (6.6), as the number of iterations approaches infinity the gain converges to about 1.65. During actual implementation of the circuit the multiplication can be removed by increasing the bit size of the output of the CORDIC circuit to accommodate the gain or to have the gain corrected for if desired by manipulating the gains of subsequent filters in the design,

$$G_{cordic} = \prod_{k=2}^{n} \frac{1}{cos(\theta_k)}, \tag{6.6}$$

where k is the iteration number and n is the total number of iterations used in the CORDIC circuit.

The CORDIC can be converted into a NCO by replacing the ROM in figure 6.7 with the CORDIC function and placing a constant value into one of the CORDIC inputs and a zero into the other.

Furthermore because the CORDIC ultimately performs the vector rotation given in (6.3), the received signal at the demodulator can be fed directly into the y input of the CORDIC based NCO as shown in the block diagram in figure 6.8. If this is done the cosine and sine outputs of the CORDIC downconverter yield 2.12 and 2.13 which gives the correctly demodulated I and Q signals. The use of the CORDIC, in this case, saves the two multipliers that are used for the downconversion process done in more traditional receivers such as the one presented in [26].

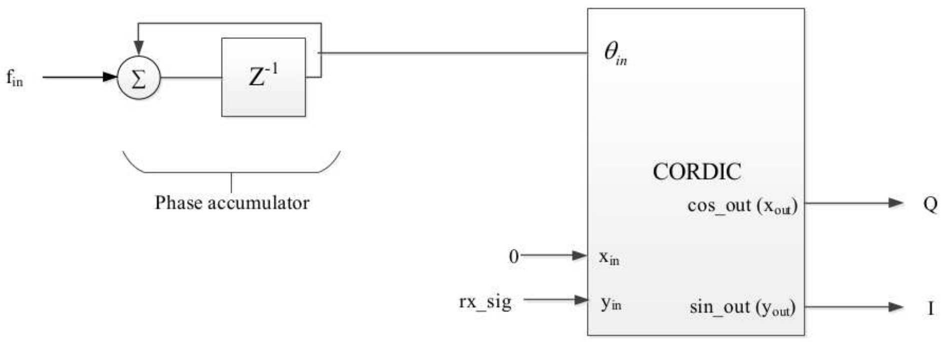


Figure 6.8: Block Digram of CORDIC downconverter

6.4.2 Specification and Design of the CORDIC Downconverter

The DOCSIS 3.0 specifies that a frequency resolution of 1 Hz that needs to be supported in order to satisfy the frequency offset command. The command is used as a fine adjustment in the event that there is some frequency mismatch between the CM and the CMTS. This 1 Hz resolution dictates the number of bits that must be used in the phase accumulator of the CORDIC based downconverter according to (6.7),

$$N_{phase} = -log_2\left(\frac{\Delta f}{F_s}\right), \tag{6.7}$$

where Δf is the frequency resolution required and F_s is the sample frequency of the NCO.

In addition to the frequency resolution requirement the DOCSIS also gives a requirement for phase noise produced by any NCOs in the system. The phase noise requirement is for at least -46 dBc. That is to say that any spurious emissions caused in the NCO must be at least 46 dB below that of the carrier signal.

The amount of reduction of the noise and spurious emissions caused by the CORDIC NCO is dependent upon of the number of bits used in the x and y data paths of the CORDIC, the number of iterations done in the CORDIC and the number of bits used in the phase accumulator of the NCO.

In order to achieve the 1 Hz resolution required by the DOCSIS a phase accumulator of 27 bits was used in the design of the CORDIC down-converter according to (6.7). The data width and number of iterations (stages) used in the CORDIC NCO were selected to be 18 bits each. The reason behind the selections was to produce 18 bit accurate results for the output of the CORDIC so that the 18-bit by 18-bit multipliers used in subsequent filters would be fully utilized. The selection of 18 stages for the CORDIC circuit results in a gain of 1.647 according to (6.6). This gain requires an additional bit to be used in the CORDIC circuit's internal signals for the x and y values, unless a multiplier is used to first scale the input values by the reciprocal of the gain. In order to save multipliers this circuit has been designed to include the extra bit in the data paths internal to the circuit. The output of the CORDIC circuit is then truncated down to 18 bits, from the 19-bit internal signals used.

The structure of the first two stages (iterations) used to implement the CORDIC block of the down-converter are shown in figure 6.9. The remaining stages used in the CORDIC down-converter are simply copies of the second stage shown with an extra bit shift to the right for each additional stage. These bits shifts have been denoted by a sign extender in the figure indicating a sign extension by N-1 bits, where N is the stage number. It should be noted that the first stage is slightly different than subsequent stages used in the CORDIC circuit. The first stage does a rotation by ninety degrees by taking the twos complement of the x or y inputs depending upon the sign of the phase input, θ_{in}, at the first stage.

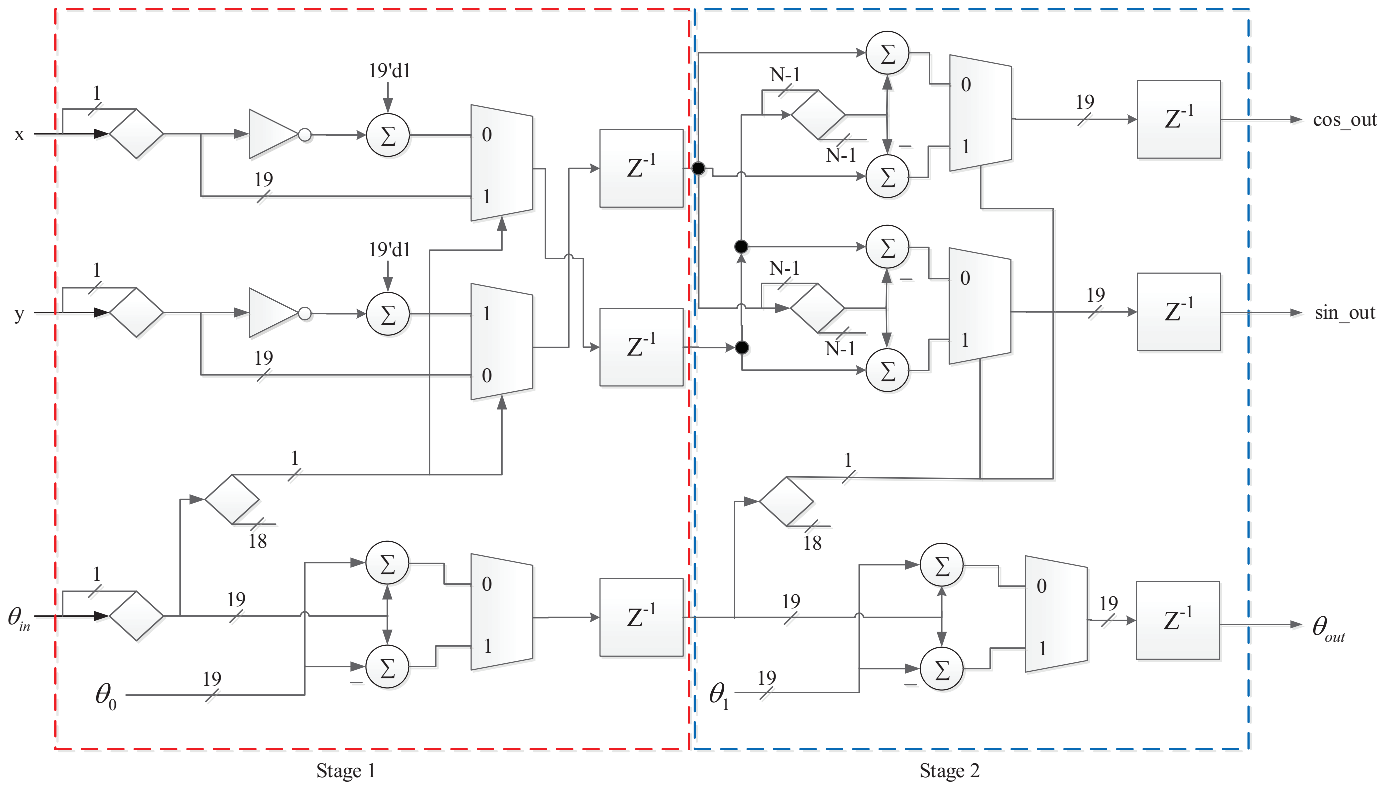


Figure 6.9: First Two Stages of CORDIC circuit

6.4.3 Testing of the CORDIC downconverter

The CORDIC down-converter was written in verilog HDL and placed onto an FPGA. The FPGA was then connected to a Signal Analyzer and oscilloscope for testing purposes. Using the in-system probes and sources feature of the ALTERA Quartus tools the frequency input to the phase accumulator was varied manually to verify the frequency resolution of the down-converter as less than 1 Hz.

A swept frequency was then placed as an input to the phase accumulator in order to test the frequency range of the CORDIC based NCO and to ensure that any noise produced met the DOCSIS requirements.

The initial tests were performed with a constant input value placed into the x input of the circuit and a zero value placed into the y input.

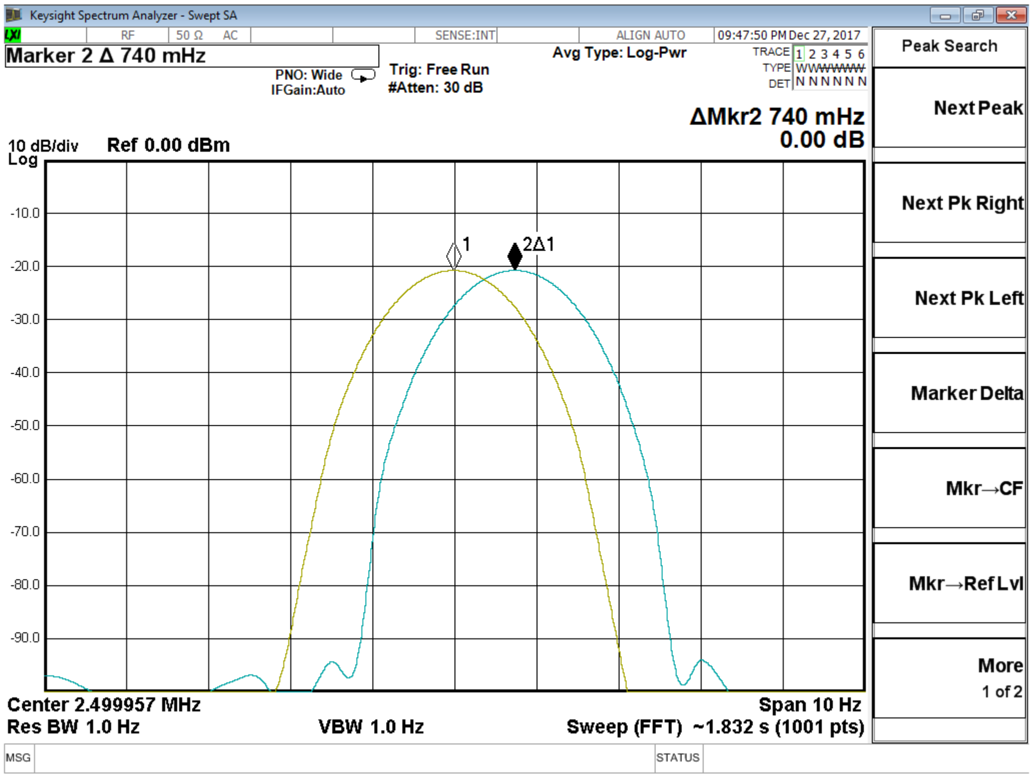
The figure shown in 6.10 shows the minimum frequency change when increasing the input to the phase accumulator by one bit. The test was done by setting the input to the phase accumulator to a constant value holding then capturing the resulting output using a Keysight CXA N9000A signal analyzer and then increasing the constant value by one and capturing a second trace at the new frequency. The difference between the frequencies was then taken to test the minimum frequency resolution of the CORDIC NCO. The test results closely agree with the theoretical result of 0.745 Hz.

Figure 6.11 shows the full frequency range of the CORDIC circuit. The roll-off of the output seen in the figure is well known and expected. This roll-off is due to the sample and hold nature of the digital to analog converter (DAC) used on the FPGA expansion board and can be modeled theoretically with the use of (6.8) [23]. The figure was generated by adjusting the input to the phase accumulator of the CORDIC NCO by a constant value at a constant rate,

$$H_{DAC} = \frac{sin(\pi x)}{\pi x}, \tag{6.8}$$

where x is the frequency, in cycles per sample, being attenuated by the DAC sample and hold characteristic.

To verify that the output of the circuit produced both sine and cosine components the



```
RF   50 Ω   AC        SENSE:INT        ALIGN AUTO   09:47:50 PM Dec 27, 2017
Marker 2 Δ 740 mHz                     Avg Type: Log-Pwr     TRACE 1 2 3 4 5 6
              PNO: Wide    Trig: Free Run                    TYPE  WWWWWW
              IFGain:Auto  #Atten: 30 dB                     DET   N N N N N N
                                                      ΔMkr2 740 mHz
10 dB/div   Ref 0.00 dBm                                    0.00 dB
Log
Center 2.499957 MHz                                         Span 10 Hz
Res BW 1.0 Hz          VBW 1.0 Hz        Sweep (FFT)  ~1.832 s (1001 pts)
```

Figure 6.10: Test for minimum frequency adjustment of CORDIC NCO

two outputs of the circuit were sent to the DACs on the expansion board of the FPGA and connected to a Keysight DSO-X 2022A oscilloscope and the phase difference between the two signals were measured. The results of the measurement can be seen in figure 6.12. It is clear from these results that the CORDIC circuit does indeed generate the correct phase between the two signals.

To ensure that the CORDIC upconverter and downconverters used in the modulator and demodulator circuits were able to upconvert and downconvert the pulse-shaped data signal, the I and Q portions of the data signal were fed into the x and y inputs of the CORDIC based upconverter along with a constant frequency value. The cos_out output of the upconverter was then fed to one of the DACs on the FPGA daughter board and displayed on a signal analyzer, the results of which are shown in figure 6.13.

The cosine output of the modulator was then digitally fed into the x input of the CORDIC demodulator and the cos_out output of the demodulator was fed into the other DAC on the FPGA daughter board and examined on a signal analyzer to ensure that the signal was

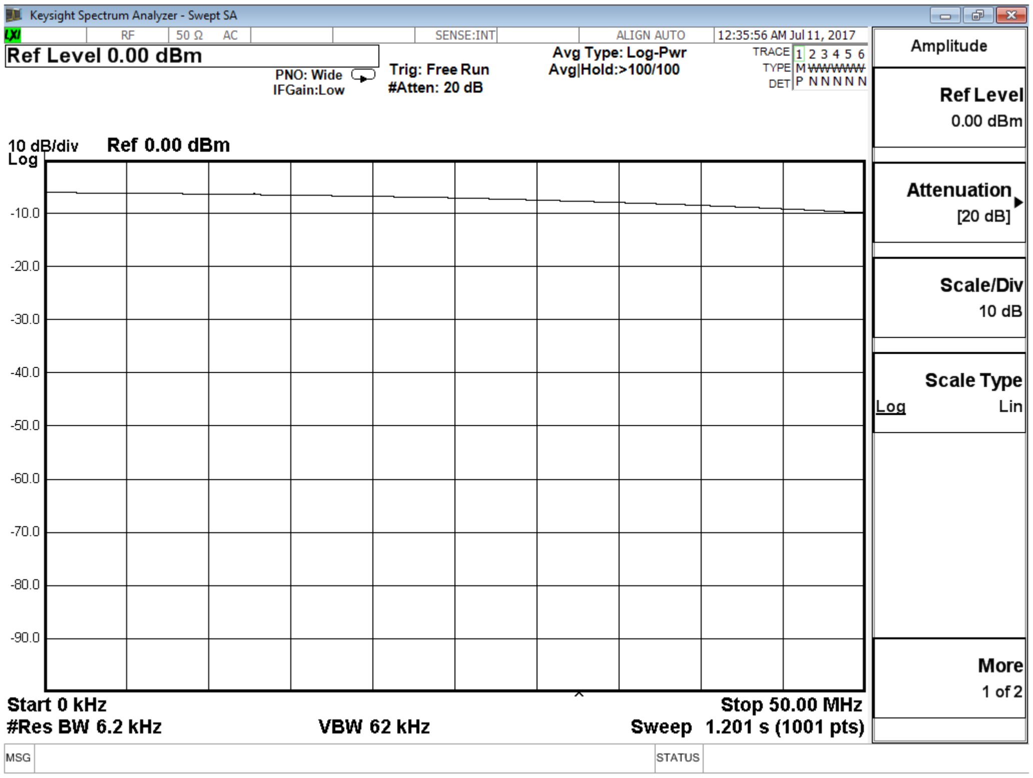


Figure 6.11: Sweep test of CORDIC NCO run at 100 MHz

downconverted to base-band as expected. The result of the downconversion process is shown in figure 6.14.

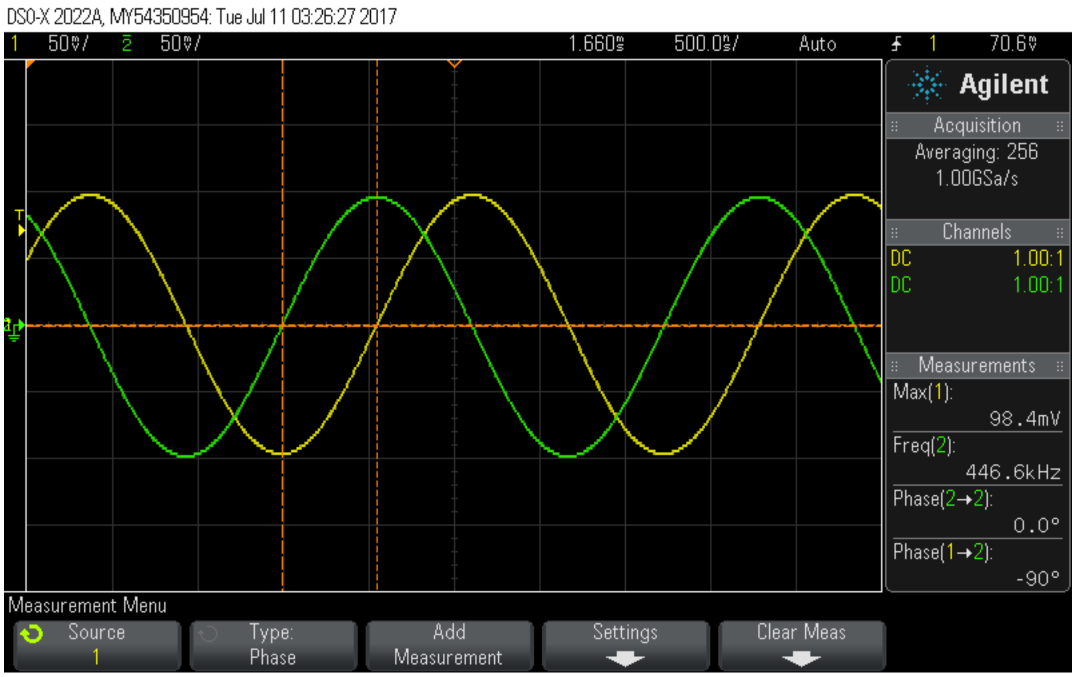


Figure 6.12: Quadrature outputs from CORDIC

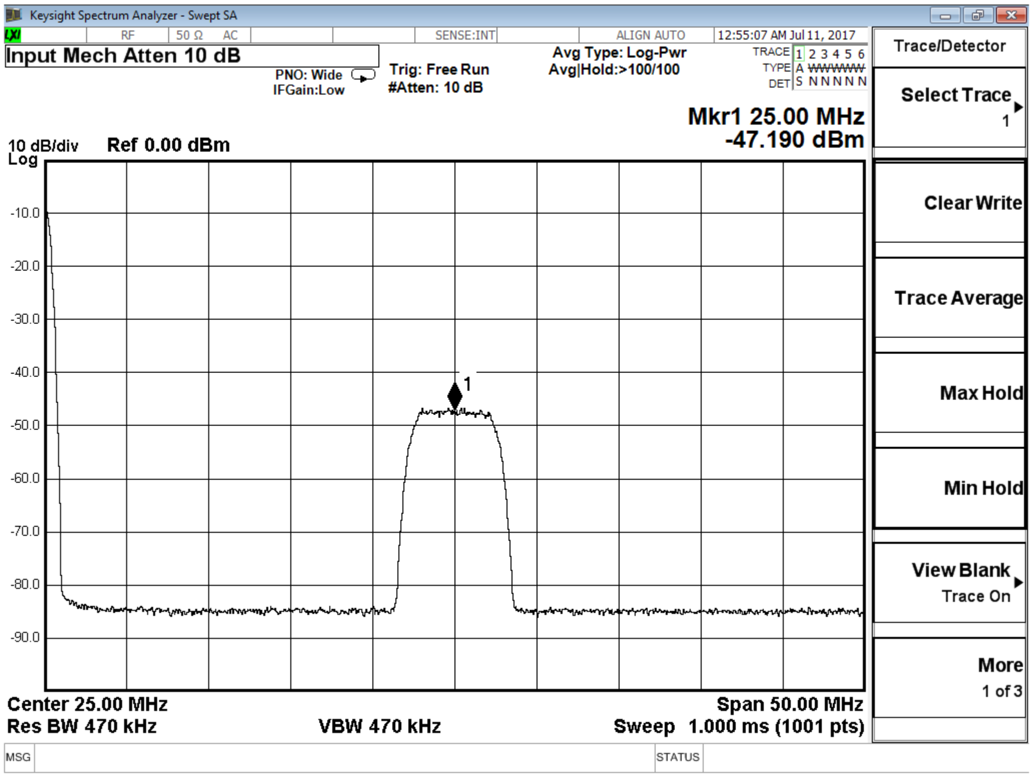
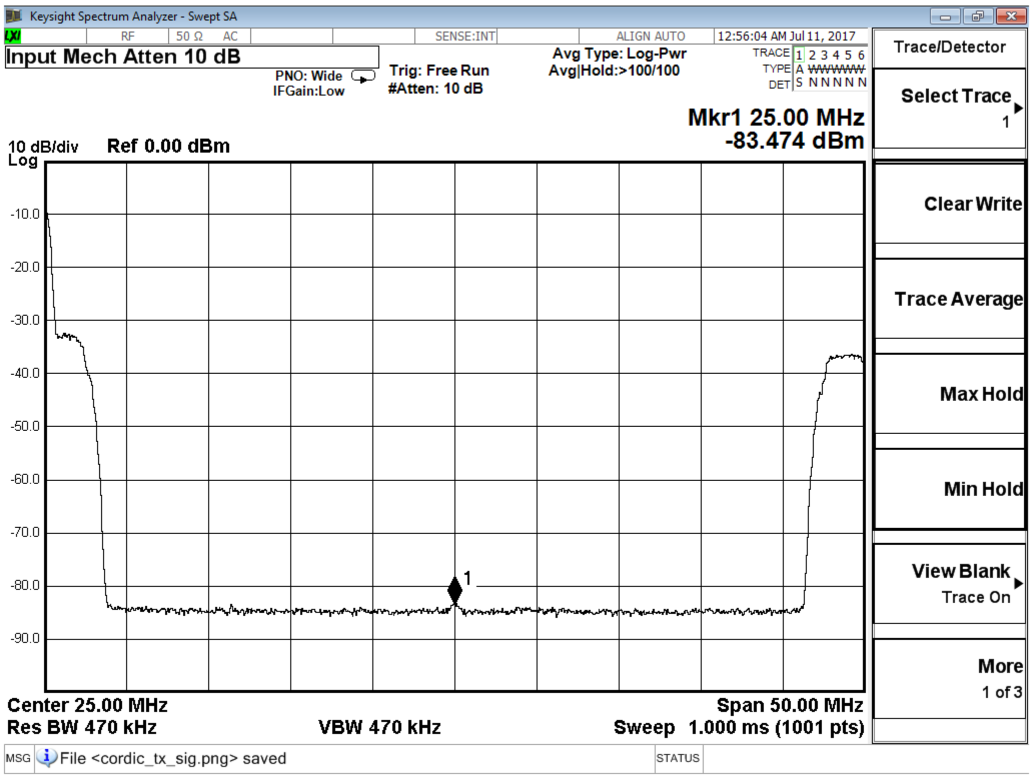

Keysight Spectrum Analyzer - Swept SA
RF 50 Ω AC SENSE:INT ALIGN AUTO 12:55:07 AM Jul 11, 2017
Input Mech Atten 10 dB
PNO: Wide IFGain:Low
Trig: Free Run #Atten: 10 dB
Avg Type: Log-Pwr Avg|Hold:>100/100
TRACE 1 2 3 4 5 6 TYPE A WWWWW DET S N N N N N
Mkr1 25.00 MHz -47.190 dBm
10 dB/div Log Ref 0.00 dBm
-10.0
-20.0
-30.0
-40.0
-50.0
-60.0
-70.0
-80.0
-90.0
1
Center 25.00 MHz
Res BW 470 kHz
VBW 470 kHz
Span 50.00 MHz
Sweep 1.000 ms (1001 pts)
MSG
STATUS
Trace/Detector
Select Trace 1
Clear Write
Trace Average
Max Hold
Min Hold
View Blank Trace On
More 1 of 3

Figure 6.13: CORDIC modulator output


Keysight Spectrum Analyzer - Swept SA

RF | 50 Ω AC | SENSE:INT | ALIGN AUTO | 12:56:04 AM Jul 11, 2017

Input Mech Atten 10 dB

PNO: Wide
IFGain:Low

Trig: Free Run
#Atten: 10 dB

Avg Type: Log-Pwr
Avg|Hold:>100/100

TRACE 1 2 3 4 5 6
TYPE A WWWWW
DET S N N N N N

Mkr1 25.00 MHz
-83.474 dBm

10 dB/div Ref 0.00 dBm
Log

Center 25.00 MHz
Res BW 470 kHz
VBW 470 kHz
Span 50.00 MHz
Sweep 1.000 ms (1001 pts)

MSG File <cordic_tx_sig.png> saved
STATUS

Figure 6.14: CORDIC downconverter output

6.5 Antialiasing Filters

As previously explained the filter structure used to implement the antialiasing filters was chosen to consist of M-path all-pass recursive filters. It was determined that the filter could be designed by exclusively using first order all-pass filter sections. The structure of the first order all-pass section used for implementation is shown in figure 6.15.

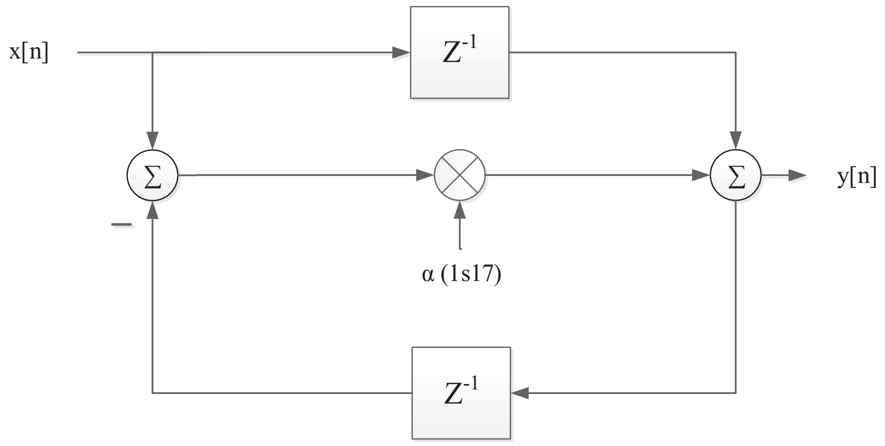


Figure 6.15: First Order All-pass Filter Section

When practically implementing the filters the signal gains throughout the filters must first be determined. The differential system equation for the all-pass structure in figure 6.15 is given in (6.9),

$$y[n] = x[n-1] + \alpha x[n] - \alpha y[n-1], \tag{6.9}$$

where α is the first order all-pass section coefficient.

Converting (6.9) to the z-domain the transfer function of the system can be shown to be (6.10), and that the magnitude response of the section is unity regardless of the value of α as shown in (6.11).

$$H(z) = \frac{\alpha + z^{-1}}{1 + \alpha z^{-1}} \tag{6.10}$$

$$\begin{aligned}|H(e^{j\omega})| &= \sqrt{H(e^{j\omega})^{*}H(e^{-j\omega})} \\ &= \sqrt{\left(\frac{\alpha + e^{j\omega}}{1+\alpha e^{j\omega}}\right)\frac{\alpha + e^{-j\omega}}{1+\alpha e^{-j\omega}}} \\ &= \sqrt{\frac{1+\alpha^2+\alpha(e^{j\omega}+e^{-j\omega})}{1+\alpha^2+\alpha(e^{j\omega}+e^{-j\omega})}} \\ &= 1\end{aligned} \tag{6.11}$$

Given that the input to the antialiasing filter is received from the output of the CORDIC downconverter, the input to the antialiasing filter can have a magnitude as large as 1.65. The output of each of the allpass sections in the first part of the antialiasing filter can therefore also have a magnitude as large as 1.65. This would indicate that the signal formats used for both the input and output formats of the filter sections can be the same.

Since the all-pass structures have a feedback path involved, the internal signals may have a gain associated with them that may be different than both the input and output signal sizes. To determine the internal signal gain of the first order all-pass structure being used, the gain at the input of the multiplier must be examined relative to the input signal. The transfer function for the all-pass structure from the input of the multiplier can be shown to be (6.12) and the square of the magnitude response to be (6.13). Substituting the trigonometric identity in (6.14) into (6.13) yields the equation for the internal signal gain given in (6.15).

Since the coefficient values for α for a first order all-pass must be real and have a magnitude less than 1 and the maximum value of the cosine evaluates to 1, equation (6.15) will be less than or equal to 2 for all values of α and ω. It becomes clear that for all cases of α and ω it would be best to design the input to the multipliers of the all-pass sections with a sign extension of one bit to accommodate a gain of two.

$$H(z) = \frac{1-z^{-2}}{1+\alpha z^{-1}} \tag{6.12}$$

$$\begin{aligned}|H(e^{j\omega})|^2 &= \left(\frac{1-e^{-2j\omega}}{1+\alpha e^{-j\omega}}\right)\frac{1-e^{2j\omega}}{1+\alpha e^{j\omega}} \\ &= \frac{2-e^{-2j\omega}-e^{2j\omega}}{1+\alpha^2+\alpha(e^{-j\omega}+e^{j\omega})}\end{aligned} \tag{6.13}$$

$$cos(\omega) = \frac{e^{j\omega} + e^{-j\omega}}{2} \tag{6.14}$$

$$\begin{aligned} |H(e^{j\omega})| &= \sqrt{|H(e^{j\omega})|^2} \\ &= \sqrt{\frac{2 - 2cos(2\omega)}{1 + \alpha^2 + 2\alpha cos(\omega)}} \end{aligned} \tag{6.15}$$

The antialiasing filter being designed consists of 2-path, 3-path and 5-path near linear phase filter stages. The input format of the first stage must be a 2s16 format to accommodate the CORDIC gain from the downconverter.

The 2-path stage block diagram is shown in figure 6.16. Since both paths of the 2-path stage are added together the the section can have an end gain of 2. Since the internal gains of the all-pass first order sections are at most two, it makes sense to sign extend the input to the 2-path stage to a 3s15 signal and use this format throughout the entire 2-path stage of the filter. At the end of the 2-path stage the gain of two, from the addition of the two filter paths, can be removed without the use of a multiplier. A division by a power of two can simply be implemented by bit shifting the word to the right. Furthermore, no bit shift is required if the 3s15 output is simply treated as a 2s16 output instead implementing a pseudo bit shift.

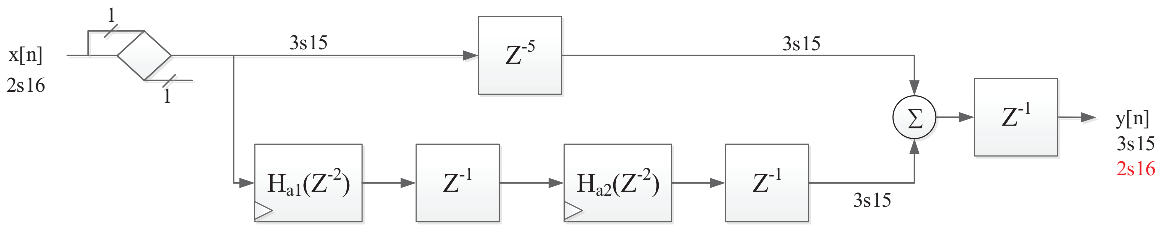


Figure 6.16: 2-Path All-pass Filter Stage

The 3-path stage results in a gain of 3 due to the addition of the 3 all-pass filter paths. The gain from the path addition results in a maximum output of 4.95; the gain of the stage multiplied by the CORDIC gain. This stage will therefore require two extra bits at the output to accommodate the gain created throughout the filter stages to this point. Since the internal gains of the all-pass structures is still two the input signal was extended to a 3s15 signal and extended again at the output of the all-pass filter paths prior to adding the paths together to accommodate the gain of 3.

Once the 3 paths are added together the gain of 3 can be removed by the use of a multiplier with a coefficient of $\frac{1}{3}$. As hardware savings are a requirement of this design and the next 5-path stage will also require a multiplier to remove its gain, it is more efficient to use a pseudo bit-shift at the output of this stage in the same fashion as done for the 2-path stage. This pseudo bit shift effectively implements a divide by two yielding an overall gain of 1.5 from the input of the 3-path stage to the input of the final 5-path stage. This gain of 1.5 is then corrected for in the final stage of the filter.

The block diagram in figure 6.17 shows the implemented 3-path filter design with proper bit sizing.

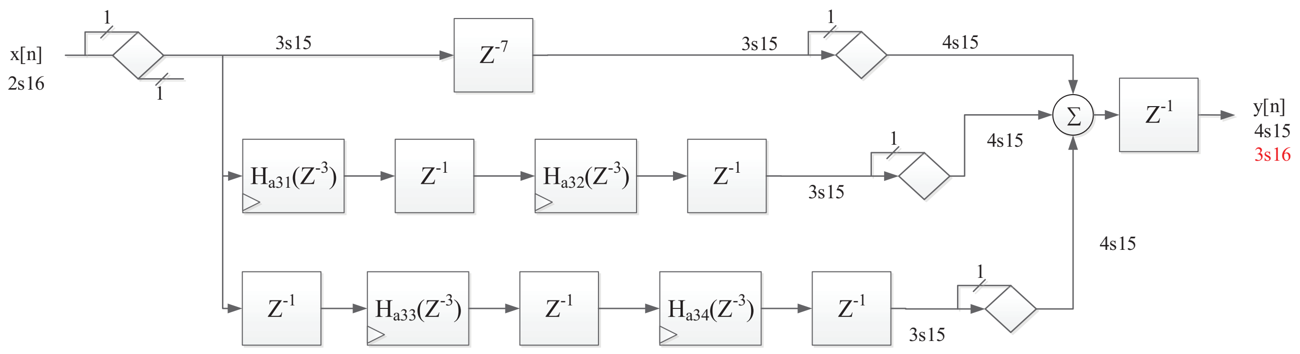


Figure 6.17: 3-Path All-pass Filter Stage

The final 5-path stage of the filter was implemented in a similar fashion to the first two stages. First the maximum input value to the 5-path stage was determined to be 2.475, requiring a signal format of 3s15. The internal gain of each path requires an extra bit at the input to each path resulting in 4s14 signal formats. The outputs from each path are then sign extended again, prior to adding them, to create a 5s14 signal to accommodate the maximum output value of 12.375 due to the gain from summing the all-pass paths in this stage of the filter. The output value of the final stage is then multiplied by the inverse of the gain through the final stage and the remaining gain through the second stage. This gain value is equal to $\frac{2}{15}$, the gain value can further be adjusted to correct for the CORDIC gain introduced during the down-conversion process yielding a correction gain of about 0.0808. In the final implementation another pseudo bit shift is used prior to the correction of the residual gains throughout the anti-aliasing filter. The bit shift provides a division by 4 causing the correction gain value to become 4 times larger and becoming 0.3232 thereby retaining better precision of the multiplier coefficient.

Since the gain gain value is less than 0.5, the signal format of the gain coefficient can be 0s18. For ease of implementation the signal formats used for the all-pass filter coefficients in the implementation of the anti-aliasing filter were chosen to be 1s17.

The final stage block diagram with signal formats is shown in figure 6.18.

Another issue that becomes relevant when sizing the filters is quantization noise effects. Each time a signal or coefficient becomes quantized to a finite precision a noise is introduced into the system. The noise is composed of both an AC and DC component that is based upon the LSB of the the truncated signal as well as the structure used for implementing the allpass filter circuits [19]. If the noise powers generated due to truncation are large enough, extra bits may be required in signal formats to accommodate the noise created by truncation. In order to ensure that the filter would preform correctly due to the truncation of signals the filter design was simulated in MATLAB with signals and coefficients truncated to the same level as described above.

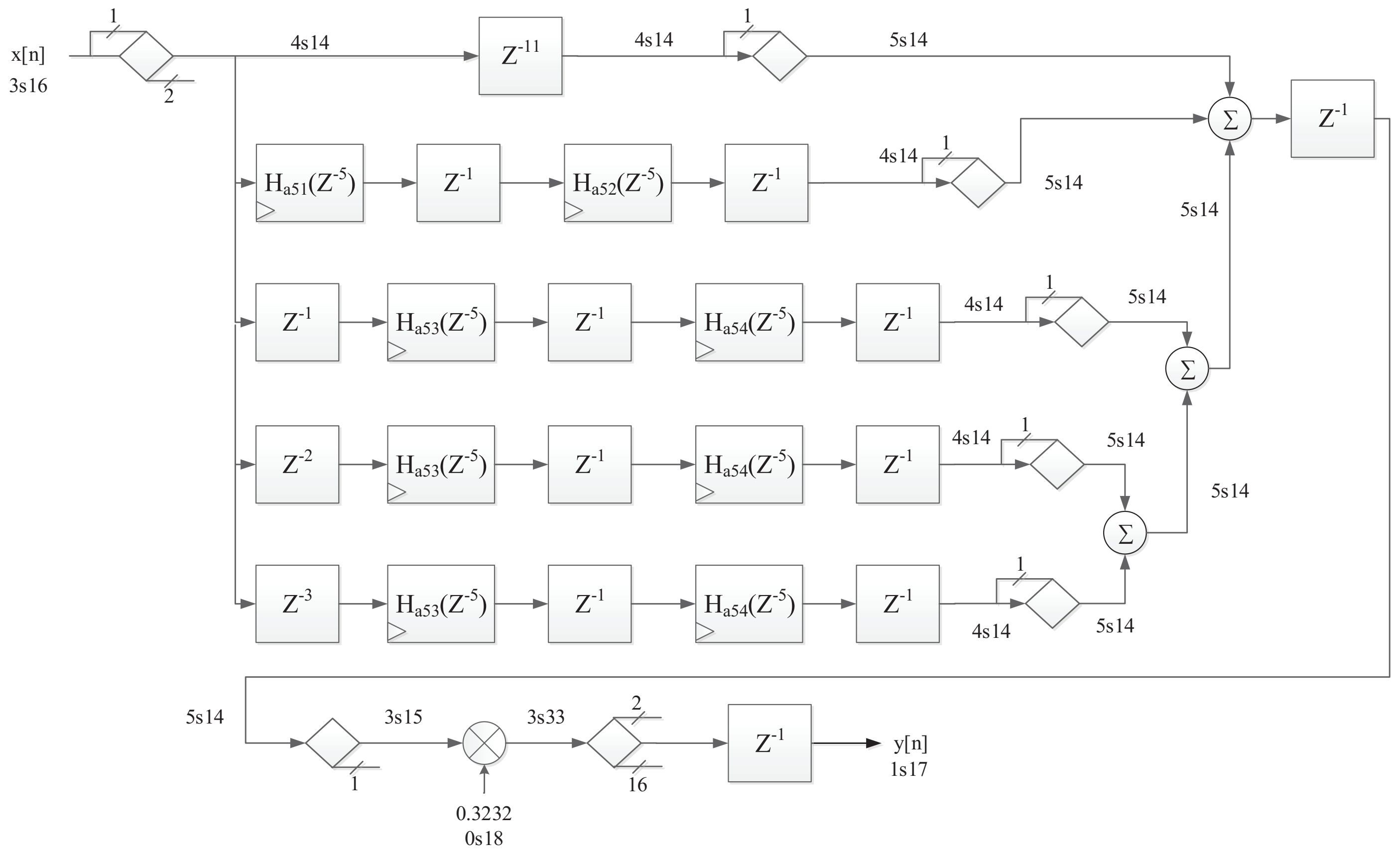


Figure 6.18: 5-Path All-pass Filter Stage

6.5.1 Multiplier Reduction

Immediately after performing anti-aliasing on the received I and Q phases a downsample is required to return the signal to the sampling rate of the matched filter. The matched filter sampling rate is five times less than the sampling rate of the antialiasing filter used in this design. Since the matched filter runs at a slower rate than that of the system clock it is possible to time-share multipliers that are run at the fastest clock rate available.

The downsample operation following the anti-aliasing filter will keep only every fifth sample from the output of the filter. Using the upsampling and downsampling identities presented in [22] the downsample operation can be moved to the input to the 5-path stage of the antialiasing filter.

Moving the downsample operation to the front side of the 5-path filter reduces the computational complexity of the filter by only needing the output of each path of the filter to be computed one fifth as often. This reduction in computation allows a single multiplier, when run at the system clock rate, to perform up to five multiplications in one period of the dowmsampled clock rate. By changing the inputs to the multiplier to use the output of the adders that feed the multipliers in the all-pass sections of the 5-path stage, the number of multipliers required in the 5-path stage are reduced from 10 to 2. The schematic diagram in figure 6.19 shows the 5-path stage redesigned to use the time-shared multiplier circuits and the schematic in figure 6.20 details the shared multiplier circuit used to implement the multiplier reduced 5-path stage. It should be noted that to have the shared multiplier circuit correctly operate, the multiplexer at the output of the circuit must have its outputs registered. The multiplier reduced stage also improves register usage by chaining the registers at the input to each path in the filter instead of having a separate delay chain for each input.

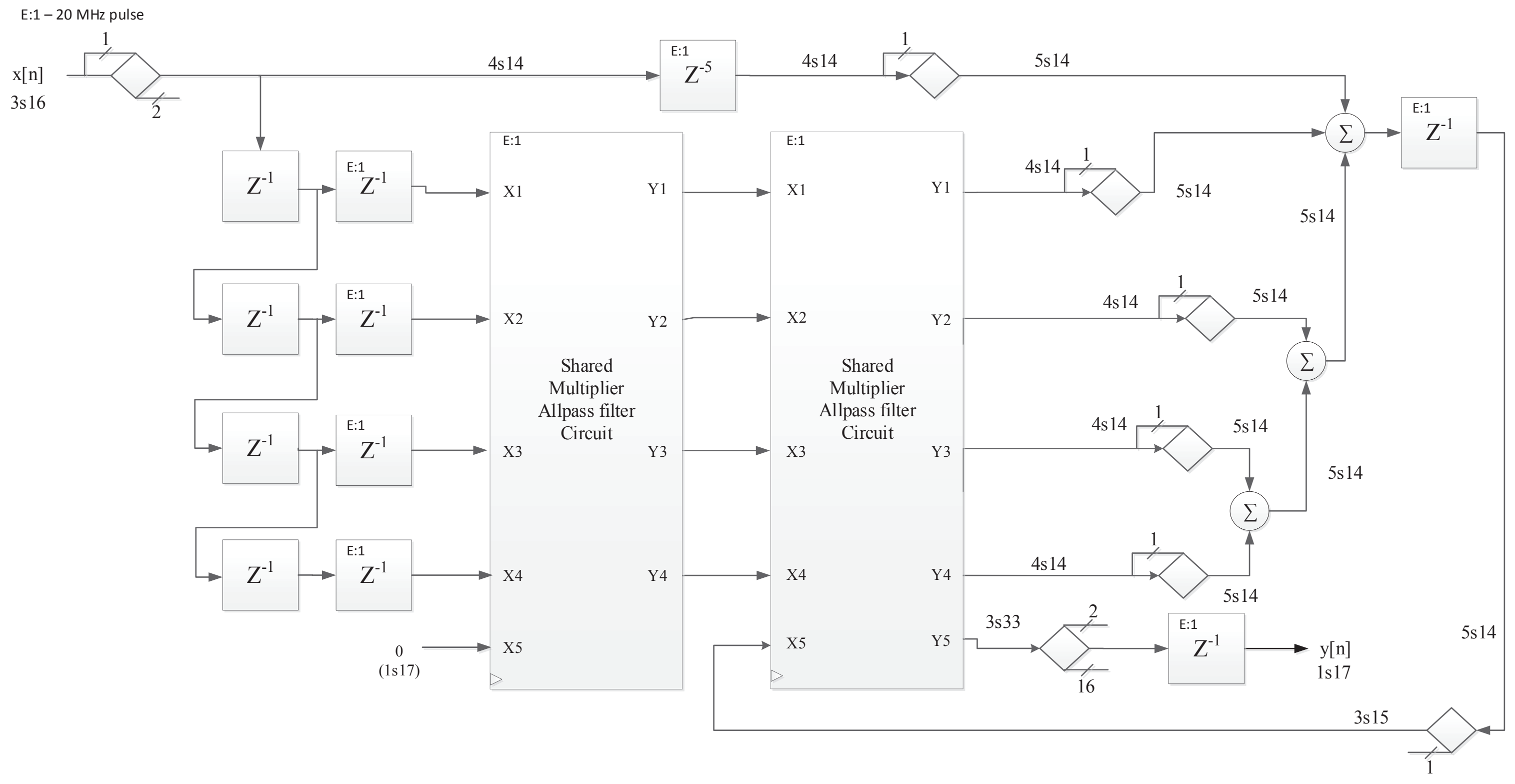


Figure 6.19: 5-Path All-pass Filter Stage with Multiplier Reduction

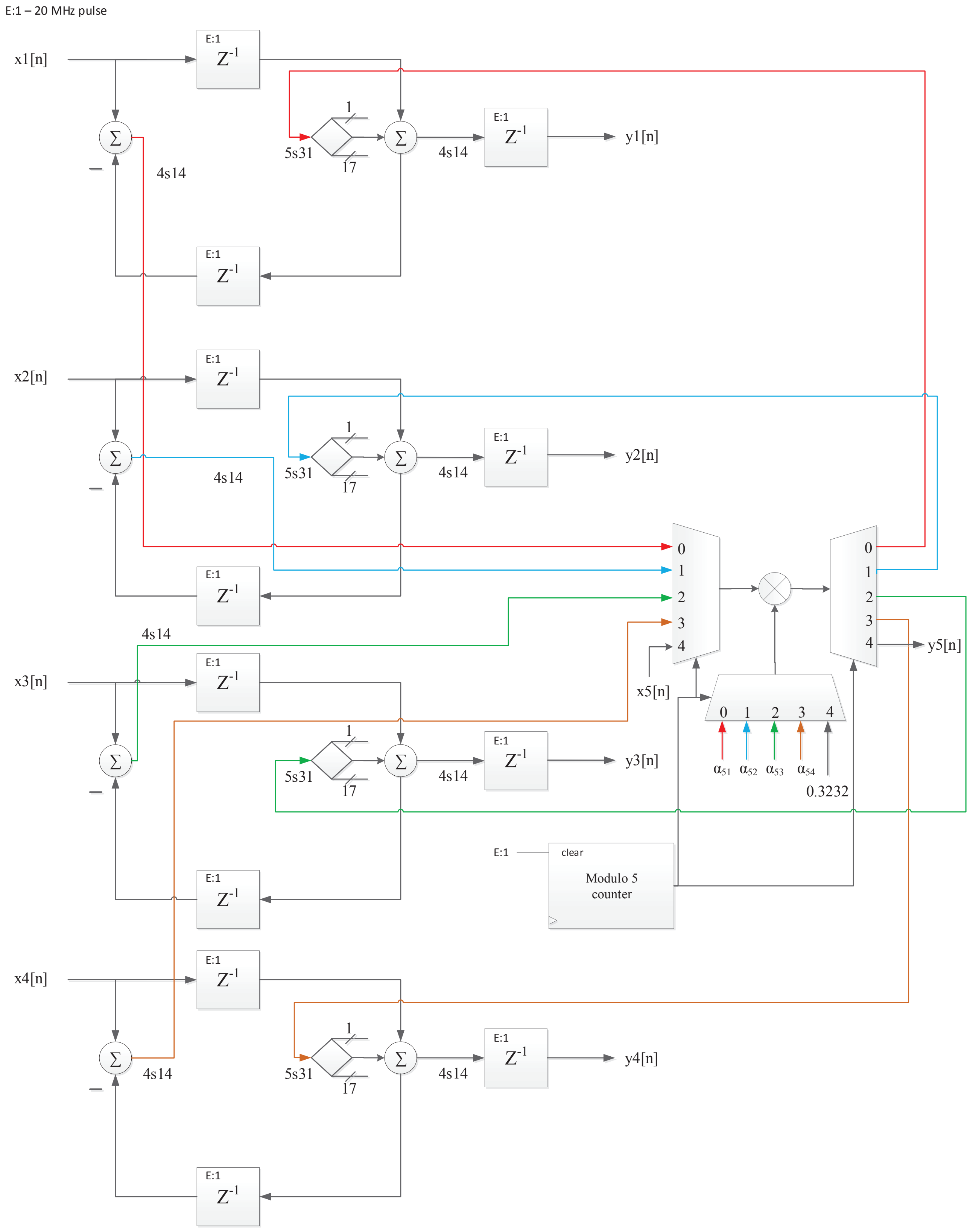


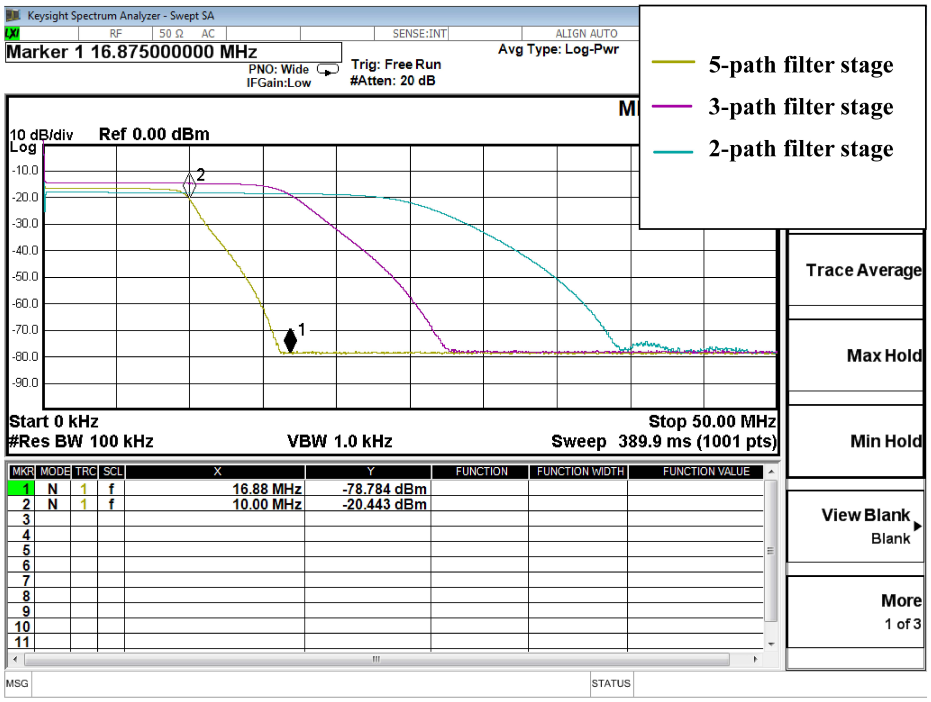
Figure 6.20: Shared Multiplier Circuit Implementation Details

6.5.2 Testing the Antialiasing Filter

For testing purposes the antialiasing filter was placed onto the DE4 FPGA and a NCO was used to generate a sinusoid with changing frequencies in 5 Hz steps from 5 kHz to 50 MHz that were fed into the input of the filter. The output of each stage of the filter was run to the DAC on the daughter board for the DE4 and the output viewed on a CXA N9000 signal analyzer to verify the correct cutoff frequency and magnitude response of the filter.

The magnitude response of each of the filters is shown in figure 6.21. The yellow, magenta and blue traces shown on the figure correspond to the output of the 5-path, 3-path and 2-path stages, respectively. The cutoff frequency of 0.1 cycles per sample is shown by marker 2 on the figure and the stop band frequency is shown by marker 1. Due to the noise floor of the signal analyzer and the daughter board being higher than the attenuation of the filter, the stop band could not be measured accurately however the shape of the magnitude response of the filters agrees with the theoretical responses discussed in chapter 5.

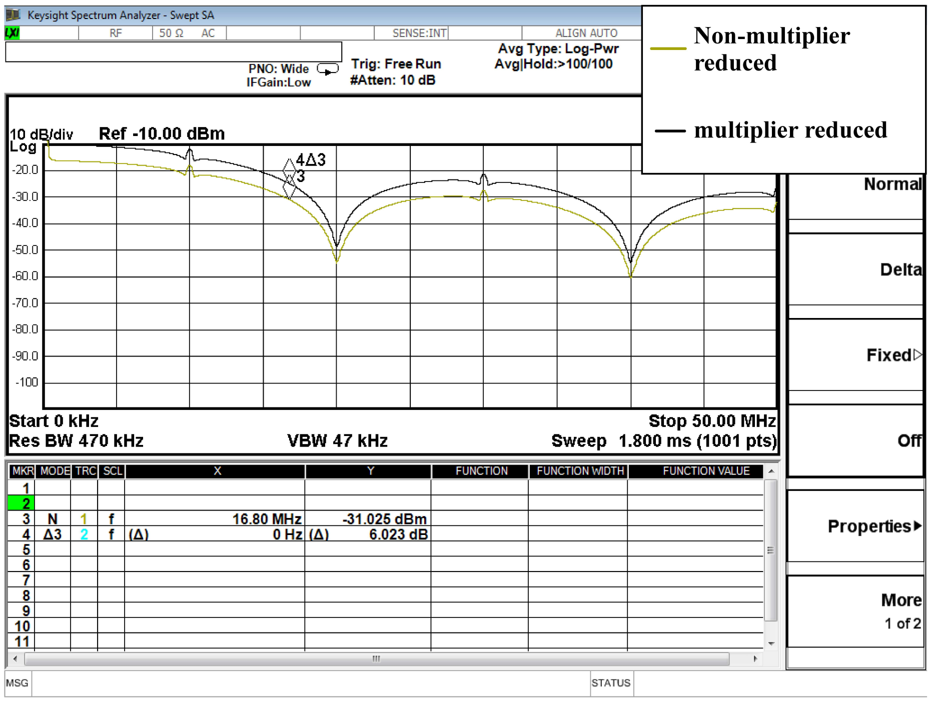
A comparison between the output of the non-multiplier reduced and multiplier reduced filter outputs was also examined on the signal analyzer. Once again, a swept frequency NCO was fed into both filters and the output of each filter was sent to the DAC and could be selected by a switch. In order to compare the two signals the multiplier reduced filter was given a gain of two on its output for clarity, resulting in the output of the multiplier reduced filter being 6 dB higher than the non-multiplier reduced filter. The output of the non-multiplier reduced filter was also downsampled by a factor of five prior to sending its output to the DAC to match the sampling rate of the multiplier reduced filter. The results shown in figure 6.22 indicate that the outputs of the two filters have the same frequency response and therefore that either of the implementation methods used will work.



Marker 1 16.875000000 MHz
PNO: Wide
IFGain:Low
Trig: Free Run
#Atten: 20 dB
Avg Type: Log-Pwr
Ref 0.00 dBm
Start 0 kHz
Stop 50.00 MHz
#Res BW 100 kHz
VBW 1.0 kHz
Sweep 389.9 ms (1001 pts)

MKR	MODE	TRC	SCL	X	Y	FUNCTION	FUNCTION WIDTH	FUNCTION VALUE
1	N	1	f	16.88 MHz	-78.784 dBm			
2	N	1	f	10.00 MHz	-20.443 dBm			
3								
4								
5								
6								
7								
8								
9								
10								
11								

Figure 6.21: Magnitude Response of the Stages of the Implemented Antialising Filter



PNO: Wide
IFGain:Low
Trig: Free Run
#Atten: 10 dB
Avg Type: Log-Pwr
Avg|Hold:>100/100
Ref -10.00 dBm
Start 0 kHz
Stop 50.00 MHz
Res BW 470 kHz
VBW 47 kHz
Sweep 1.800 ms (1001 pts)

MKR	MODE	TRC	SCL	X	Y	FUNCTION	FUNCTION WIDTH	FUNCTION VALUE
1								
2								
3	N	1	f	16.80 MHz	-31.025 dBm			
4	Δ3	2	f (Δ)	0 Hz (Δ)	6.023 dB			
5								
6								
7								
8								
9								
10								
11								

Figure 6.22: Magnitude Response of the Non-multiplier reduced and multiplier reduced Antialiaising Filters

The phase response of the filter was checked using the non-downsampled version of the filter and found to be nearly linear as expected. A phase locked loop (PLL) circuit was used to feed a sinusoid into the filter and examine the output of the filter to determine the phase offset of the filter. The results of the test are shown in figure 6.23 in blue and the theoretical phase response of a pure delay of 33 samples is shown in red. The pure delay of 33 samples was chosen to be same as the number of delays through the pure delay paths of the non-downsampled antialiasing filter plus the delay introduced by the buffers at the input and outputs of the filter in the PLL circuit.

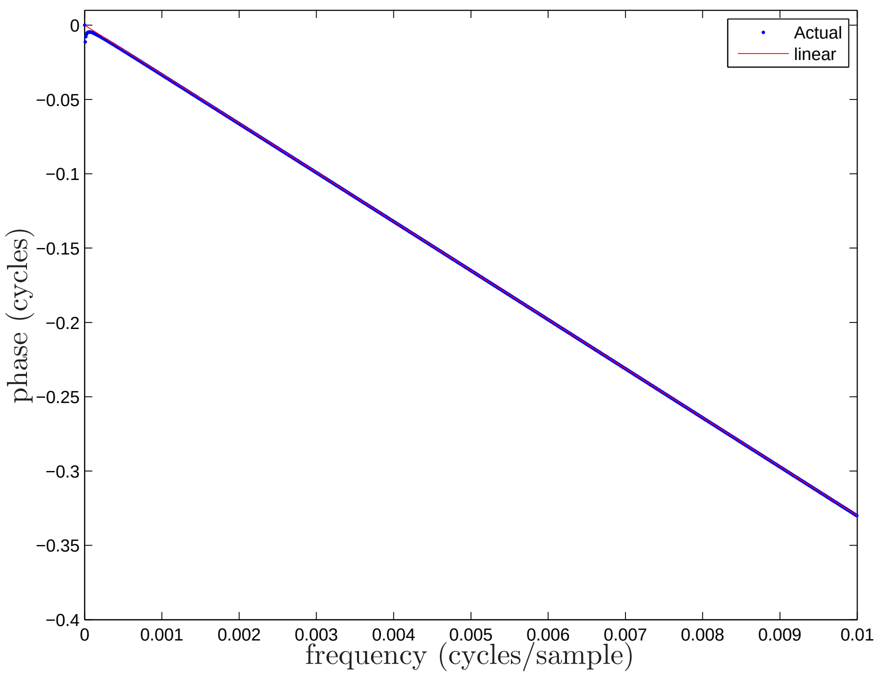


Figure 6.23: Phase response of Anti-aliasing Filter

6.6 Matched Filter

The SRRC matched filter chosen to be implemented has a length of 8 symbols, or 33 coefficients. These coefficients were determined using MATLAB and a script written to generate the filter coefficients using equation (3.1) presented in chapter 3. The timing recovery circuit design also relies on the ability of the matched filter to adjust its coefficients causing the actual implementation of the matched filter to require that multiple sets of 33 coefficients be stored in ROMs. Each set of coefficients corresponds to a fraction of a sample delay as described in chapter 5. Without going into great detail about the implementation of the timing recovery circuit at this point, it is only necessary to know that there will be 64 sets of coefficients stored for the matched filter in the ROMs corresponding to timing offsets of up to $\pm\frac{31}{64}$ of a sample.

After the coefficient sets to be used for the filter were determined the worst case gains for each set of coefficients were found. The matched filter consists of an FIR structure that will effectively have a random input signal since the data being transmitted will not be known ahead of time. To determine the worst case gain of each set of coefficients it must be assumed that the worst case gain output of the filter will be given by

$$G_{srrc} = \sum_{n=0}^{N-1} |h_{srrc}[n]|, \tag{6.16}$$

where N is the number of coefficients in each filter set. This gain assumes that each negative coefficient is multiplied by the maximum allowable negative input and each positive coefficient is multiplied by the largest positive input to the filter. In both cases the largest magnitude input to the filter is one since the magnitude of the output of the anti-aliasing filter is at most one.

Once the worst case gain of each set of coefficients was determined each coefficient set was then divided by the worst case gain found through simulation to ensure that the worst case filter output magnitude should be one.

After reducing the worst case gain of the filter to one, the size of the largest coefficient ends up being less than 0.5, allowing the coefficients to be stored as 0s18 numbers. The truncation of the coefficients in this FIR structure leads to an unwanted negative DC offset since the actual coefficients have the lower bits removed during quantization into the 0s18 format. The DC offset can be removed by adding the truncated DC amount back into the output of the filter which requires a multiplication since the amount of DC is dependent upon the input to the filter. Alternatively the DC offset can be removed by rounding the output value of the filter prior to truncation. The later method of rounding is the chosen method used in this implementation. Rounding removes almost all of the DC offset because the input and therefore output values are statistically equally negative and positive for a random signal.

The rounding implemented uses a round to zero when the fractional portion of the result is less than one half, otherwise the result is rounded to the next integer value. The round operation is performed at the output of the filter prior to the result being truncated to a 1s17 number. There are 33 multiply operations that must be performed in the filter which each produce a 36 bit output. Each of these operations must be followed by a truncation to reduce resource usage and a certain number of bits must be retained to keep enough precision in the truncated number to have accurate rounding results. Clearly for each pair of multiplier outputs that are added together prior to the output of the filter one extra bit is required to keep the number precision high enough prior to rounding. Adding the sum of each pair of multiplier outputs results in the need for another bit resulting in the need for 5 extra fractional bits in order to maintain precision throughout all the summations in the filter prior to rounding. These five extra bits cause the output of the multipliers to need to change from a 1s17 to a 1s22 number format.

The filter implementation used is shown in figure 6.24. The diagram contains blocks labeled as MF Block, each of these blocks contains the circuitry necessary for computing the product and summation of five input samples and five coefficients. In total, seven such blocks are required to implement the filter with the seventh block only computing the product and sum of three samples and coefficients. The reason for splitting the filter into these blocks is to take advantage of multiplier sharing as discussed in the next subsection.

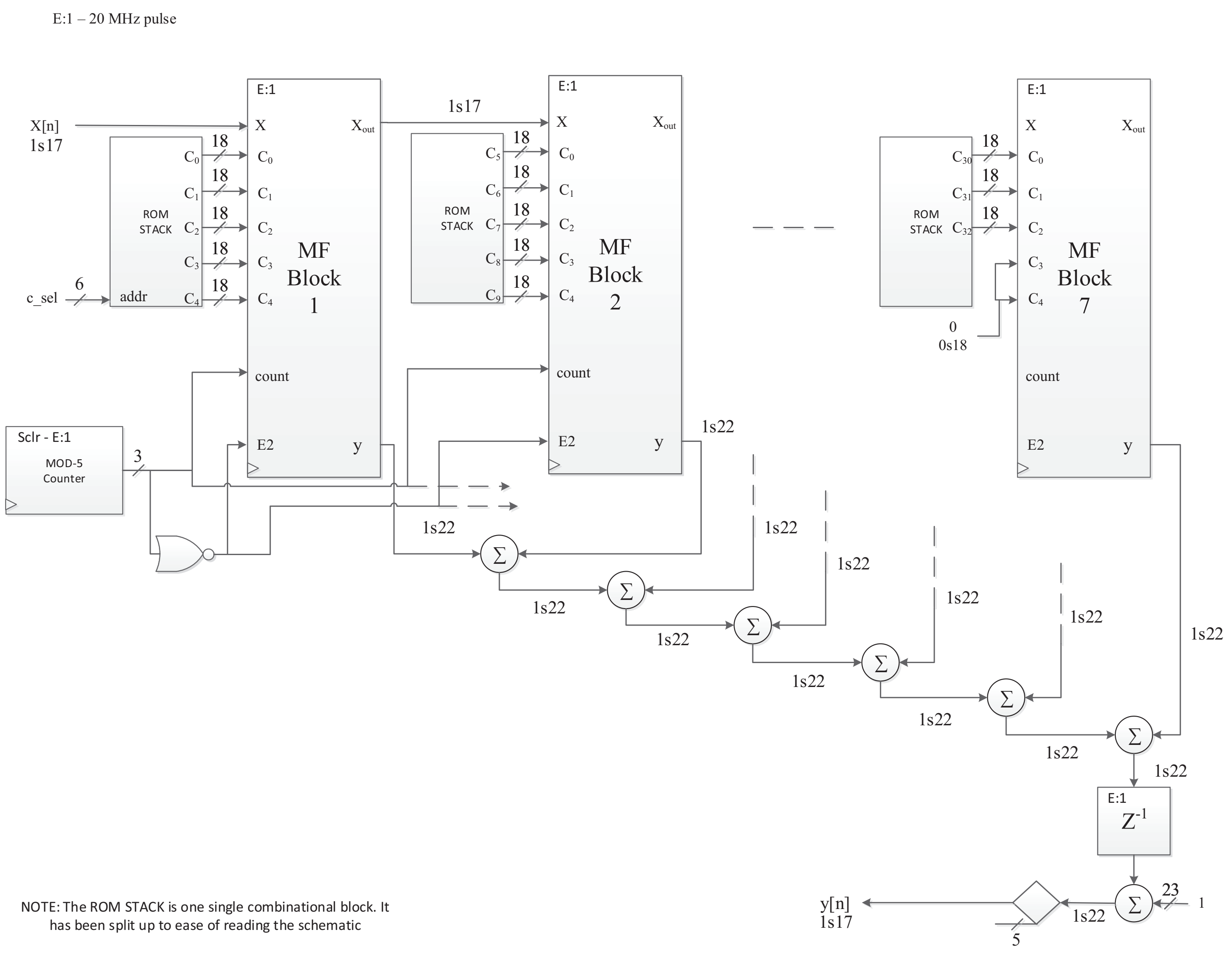


Figure 6.24: Implementation of the Matched Filter

6.6.1 Multiply and Accumulate Circuits

As with the antialiasing filter, because the system clock is run at five times the matched filter sampling rate, multipliers may be time-shared. Since FIR structures do not have feedback paths like those in the antialiasing filter, further hardware reduction may occur. FIR based filters have a structure that sums the products of samples by their respective coefficients as shown in figure 6.25, which is repeated here from Chapter 4 for clarity. Normally this is done with an adder that is used between pairs of multiplier outputs. The fact that the system clock is five times faster than that of the matched filter sampling rate, however, allows for both time-sharing the multipliers and adders in the circuit.

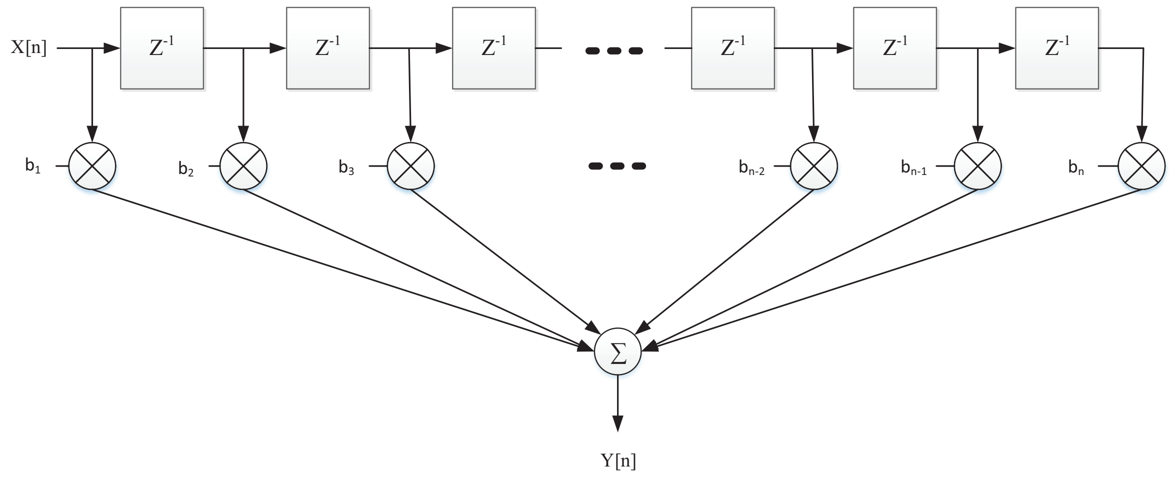


Figure 6.25: Typical FIR Filter Structure

Instead of using separate multipliers and adders for each coefficient and sample pair a circuit called a multiply and accumulate circuit can be used to reduce the number of multipliers and adders required to almost one fifth. The circuit diagram in figure 6.26 shows the details of the implementation of the multiply and accumulate circuit.

In order to accomplish five multiplications and additions in one period of the matched filter sampling rate, a counter is required. The counter counts from zero to four and then resets back to zero again. This counter is used to select the coefficient and sample that are to be multiplied together. The output of the multiplier is then fed into an accumulator circuit. The accumulator simply adds the output of the multiplier to itself at each clock edge of the system clock. When the counter rolls back over to zero the accumulator is then reset to the product of the first coefficient and first stored sample in the MF Block. At the same time the value of the accumulator prior to resetting is stored into the output register of the multiply and accumulate circuit. To ensure that the counter initializes to a correct sample time of the matched filter a synchronous clear (sclr) input is used to reset the counter. This clear signal must be tied to the sampling pulse signal of the matched filter. Using the synchronous clear results in one system clock delay in the select signal used in the multiply and accumulate circuits, this delay effect can be removed by running subsequent registers used in the matched filter at the system clock rate and using the logical nor of the counter output to enable these registers. The output of the matched filter is then placed into a register that is enabled by the matched filter sampling pulse for use in subsequent circuits circuits.

When sizing the accumulator the worst case sum of the values that could be input to the accumulator must be determined in the same fashion as that for the overall matched filter for each of the MF Blocks. From the results of these summations it was determined that no extra sign bits were required in the accumulators and as such a 1s22 signal format was sufficient.

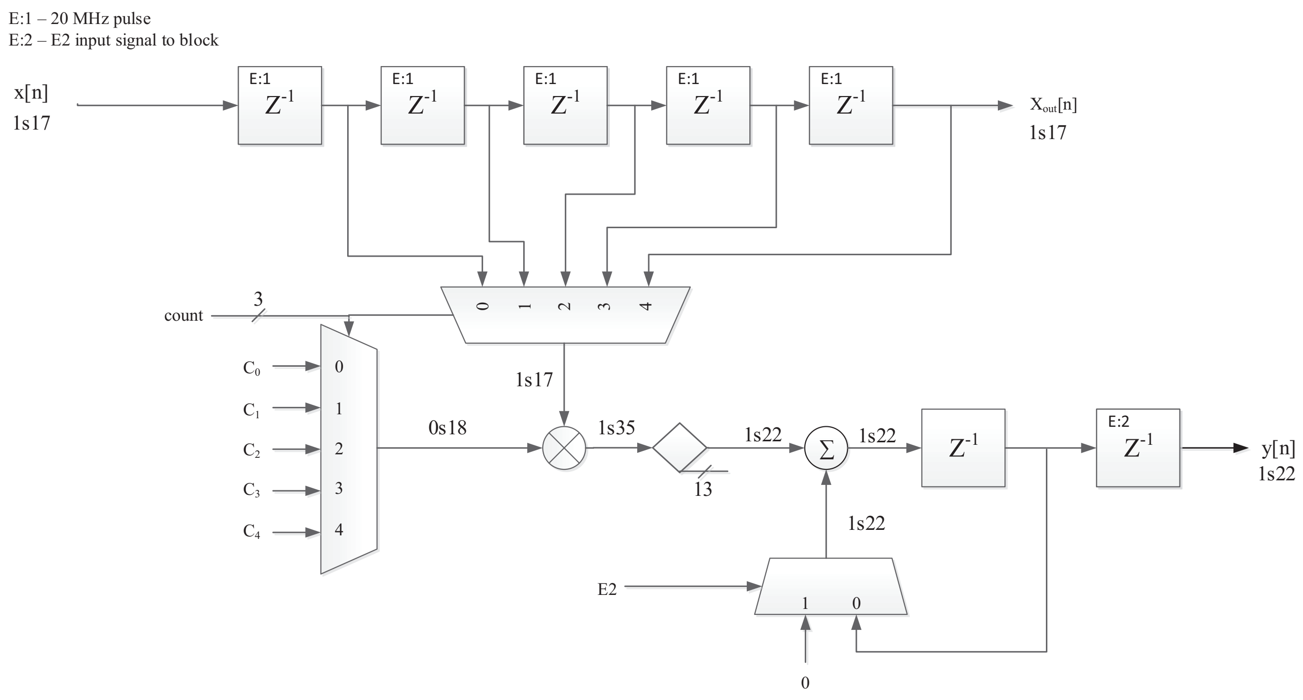


Figure 6.26: MF Block implementation with efficient multiplier usage

6.6.2 Testing

The matched filter was tested by placing a swept frequency NCO producing a full-scale 1s17 sinusoidal input into the the filter. The magnitude response of the filter was viewed on the signal analyzer and sample points stored. The sample points from the signal analyzer were then imported into MATLAB and theoretically corrected for the DAC roll-off. The corrected points were then compared to the theoretical magnitude response of the filter to show that they agreed within 0.2 dB throughout the passband. The theoretical response and the response obtained from the signal analyzer are given in figure 6.27.

A comparison between the difference in the theoretical response and the measured response is given in figure 6.28. The CXA signal analyzer used to measure the magnitude response was set to have a resolution bandwidth of 10 kHz and sampled 1001 point over the frequency range of 0 kHz to 10 MHz, the first 30 kHz of samples were ignored due to the DC marker of the signal analyzer and the fact that the DACs used do not operate well at low frequencies [3]. Outside of the passband the error plot in 6.28 has large differences, located near the zeros of the filter, due to the noise floor of the signal analyzer. The results indicate that the implementation of the matched filter presented here produces less than 0.2 dB of error throughout the passband of the filter.

The filter was then cascaded with the pulse shaping filter and fed a pseudo-random sequence generated by a 22-bit linear feedback shift register (LFSR) and passed through a 16-QAM mapper. The output of the filter was then downsampled and fed into a circuit designed to compute the MER of the cascaded filters. Signal Tap in the Quartus software suite was then used to view the computed MER. The MER determined during testing was found to be 55.5 dB, a 0.1 dB difference from the theoretical value obtained in MATLAB.

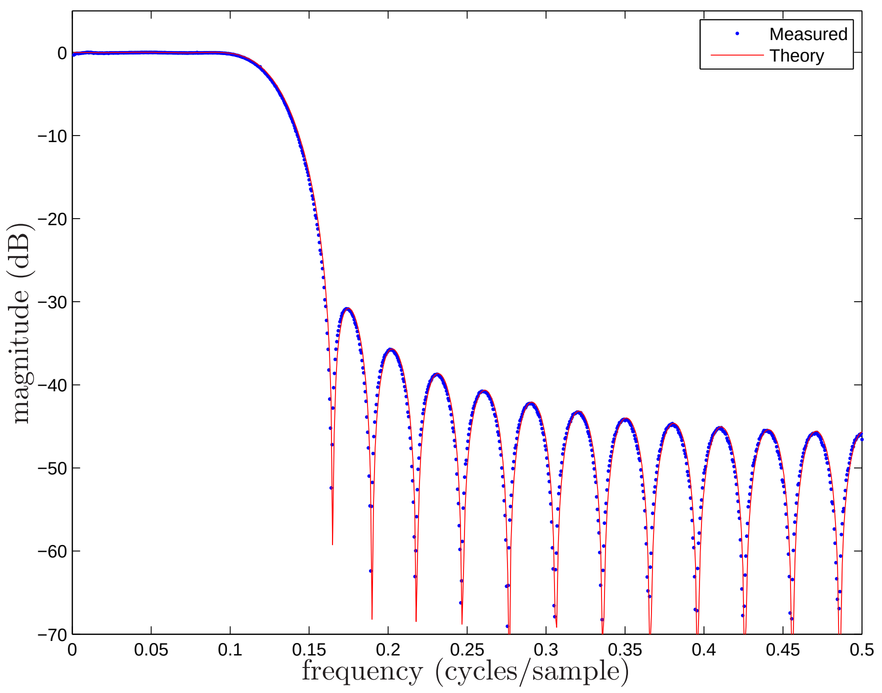


Figure 6.27: Theoretical and Measured Magnitude Response of Matched Filter

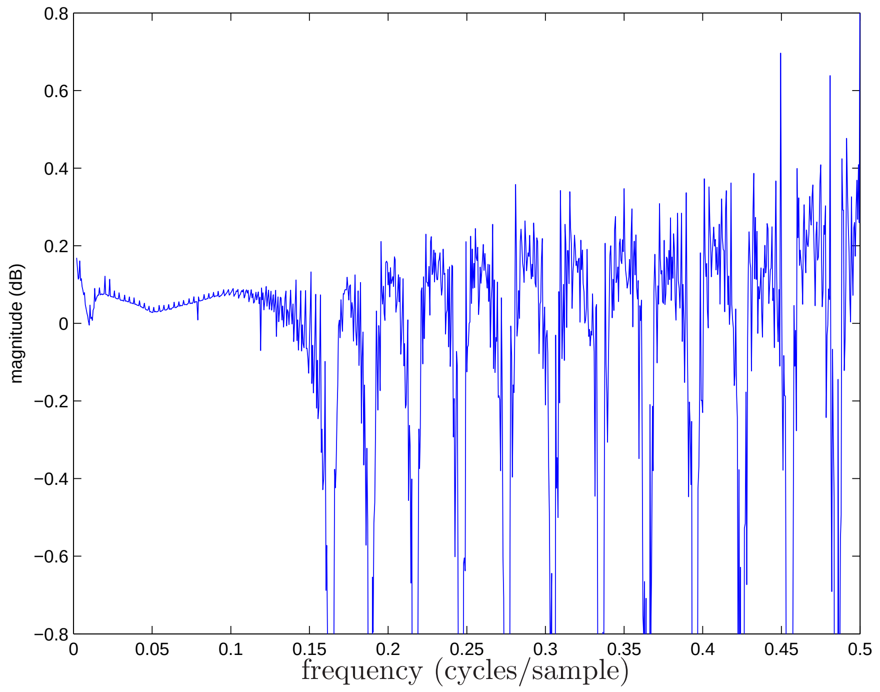


Figure 6.28: Difference between Theoretical and Measured Magnitude Response of Matched Filter

6.7 Timing Recovery

The implementation of the timing recovery circuit does not directly follow the procedure described in [17]. The circuit designed removes the AGC circuit and threshold detection requirements used in [17] and instead implements a maximum peak detection circuit at the expense of requiring the timing recovery circuit to run for at least one full preamble sequence as opposed to stopping once a peak is found that is greater than a specified threshold. The benefit in using the implementation detailed here removes the need to determine the threshold levels for the incoming signal and the reliance on a perfect AGC circuit.

This implementation relies on the MAC providing an accurate start_timing signal that indicates when the timing recovery circuit will be receiving a preamble packet. Once the MAC sends the start_timing signal the timing recovery circuit resets a counter that counts the number of samples in one preamble packet, in this case 260 samples of the system clock. Once the counter has reached 260 the counter stays at that value until the next start_timing signal is sent. The counter is also used to generate a timing_done signal to indicate that the timing recovery circuit has completed which is in turn used to signal the start of the frequency recovery circuit.

The schematic for the timing recovery block is presented in figure 6.29. The circuit is fairly large and complex, however its functions are straight forward to follow.

The output of the matched filters for the I and Q phases are input to the circuit at the signals called I_{mf} and Q_{mf}, respectively. The signals are then upsampled back to the RF rate as explained in section 5.2.4. To remove the images caused during the upsampling process the signals are then passed through low-pass filters. To speed up design time the low-pass filters used were copies of the anti-aliasing filter used in the receiver without the downsampling optimization previously discussed.

Once the upsampling images are removed the signals are passed through the timing detector circuit, shown in figure 6.30. The timing detector passes the the output of the lowpass filters through a correlation filter that determines the correlation power in each set of received samples to the known preamble sequence and sums them.

The implementation of the correlation filter is given in figure 6.31. The correlation filter yields a maximum value when the expected preamble sequence aligns with the input data to the correlation filter. Since the preamble sequence is made up of QPSK symbols with values of -1 or 1 the multipliers used in the correlation filter can be removed and replaced with adders and subtractors. Due to the correlation filter normally operating at the symbol rate it must be upsampled to the same rate as the input to the correlation filter. This upsampling does not cause issues due to images because the input signal to the filter has already been filtered to remove any signal component that would be present in the upsampled correlation filter's output.

The preamble barker sequence used is -1, -1, -1, -1, -1, 1, 1, -1, -1, 1, -1, 1, -1. Every time a -1 is expected the input value is subtracted and every time a 1 is expected the input signal is added. Each pair of subtractions or additions will cause a potential gain of 2. Since there are thirteen preamble symbols there could be a gain at the output of the correlation filter's adders of 13 requiring an extra 4 integer bits resulting in a 5s17 formatted output. The square of this value is then used to determine the power in each phase of the received signal. Since the hardware may be limited to 18 bit multipliers the 5s17 signal must first be reduced to a 5s13 signal before multiplication is performed. The output of the multiplier is then truncated to a 10s13 value.

The output of the two correlation filters are then summed and truncated to produce a 11s12 formatted signal that is used as the input to a peak detection circuit. The peak detection circuit generates a local_peak pulse every time a peak is detected at the correlation filter output. The peak detection circuit used here is very simple and produces a pulse when the following condition is met: $y[n] < y[n-1] > y[n-2]$, where y is the output of the correlation filter. This peak signal is then used as an enable to another circuit that stores the maximum peak value and its neighboring samples. These stored samples are then used as inputs to a divider circuit to determine the amount of fractional timing delay to offset the matched filter coefficients by as explained in section 5.2.4.

The matched filter was designed to have a set of 64 coefficients to implement a fractional timing delay recovery. Divider circuits generally require a large amount of resources to implement especially when they need to perform a division where the magnitude of the dividend is less than that of the divisor as is the case here since (5.7) generates values between -0.5 and 0.5. Fortunately the output of the divider will need to be multiplied by 64 to determine the correct set of matched filter coefficients and therefore the divisor can be reduced in size by 64 times using bits shifts making the magnitude of the dividend larger than the divisor. This simple step allows the divider to be implemented as an integer division. The circuit for the divider is given in Appendix A.1.

The output of the maximum peak detection circuit is also used to store the clk_phase value that occurs at the the point when the maximum peak is reached. This value indicates the clock phase counter value when the timing is correct to within one system clock sample. This value is then subtracted from the input clk_phase signal to produce a zero value indicating when a correctly timed sample should be taken. There must also be a constant added to the value to compensate for the time delays used throughout the correlation filter, lowpass filter and preceding circuitry. The constant value of 15 to be used in this circuit was determined using Signal Tap (an in-circuit logic analyzer) to view the clock phase when the maximum peak was determined and the stored clock phase value.

Additional circuitry is required to correctly produce the adj_sym_clk_ena signal that indicates the proper sampling time of incoming symbols. Since the clk_phase signal can vary between 0 to 19, which does not fit nicely into a binary number allowing for wrap-around to occur, when subtracting the stored peak clock phase and clk_phase value from one another and adding the constant value the result can have a magnitude greater than 19. To correct for this a circuit is needed to ensure that the magnitude of the result does not exceed 19. The circuit used to implement this function is shown in the lower right portion of figure 6.29.

Once the maximum peak has been detected a counter is started that is used to produce a pulse at the start of every preamble sequence for later use in the phase recovery circuit.

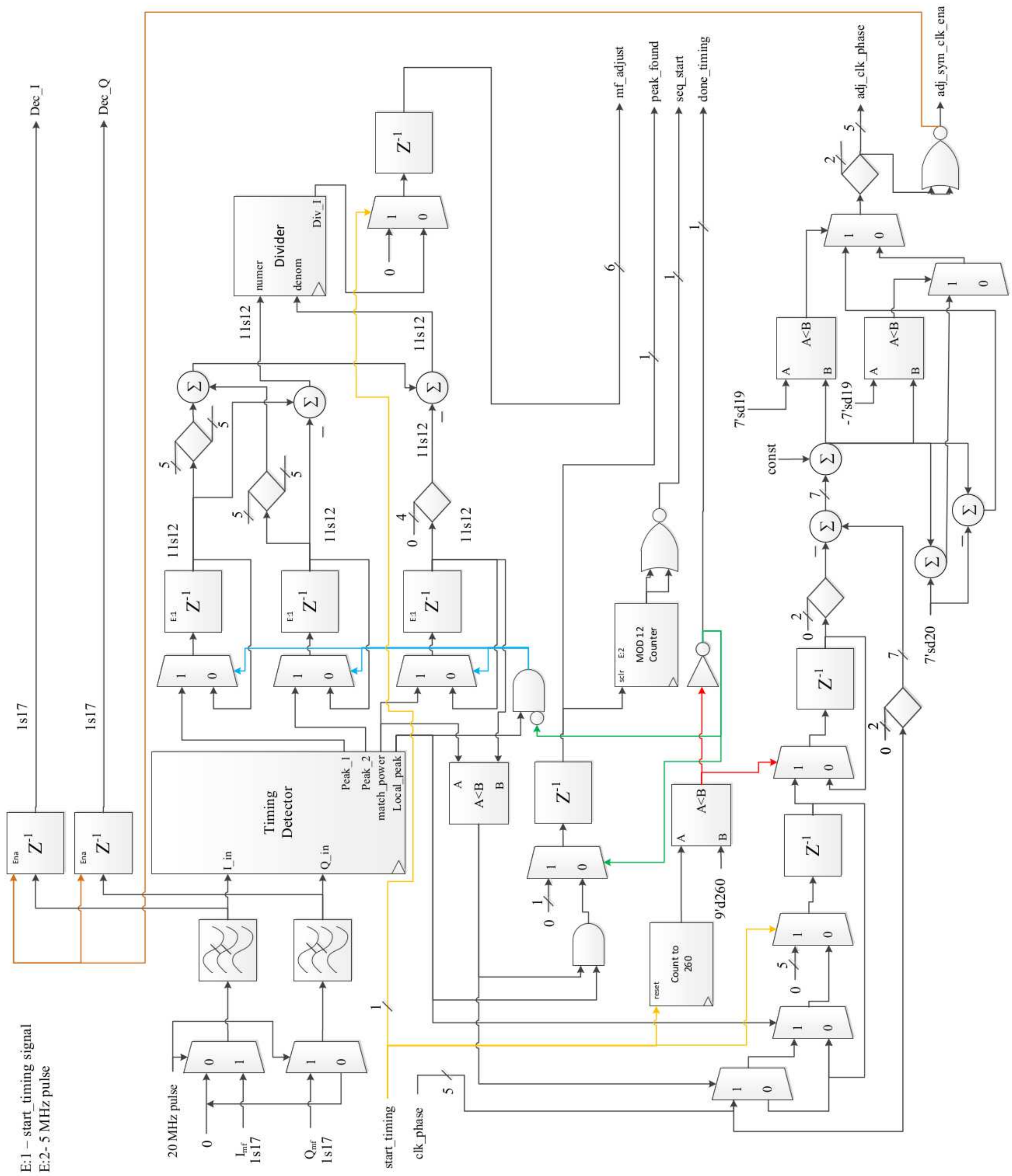


Figure 6.29: Schematic of the Timing Recovery Circuit

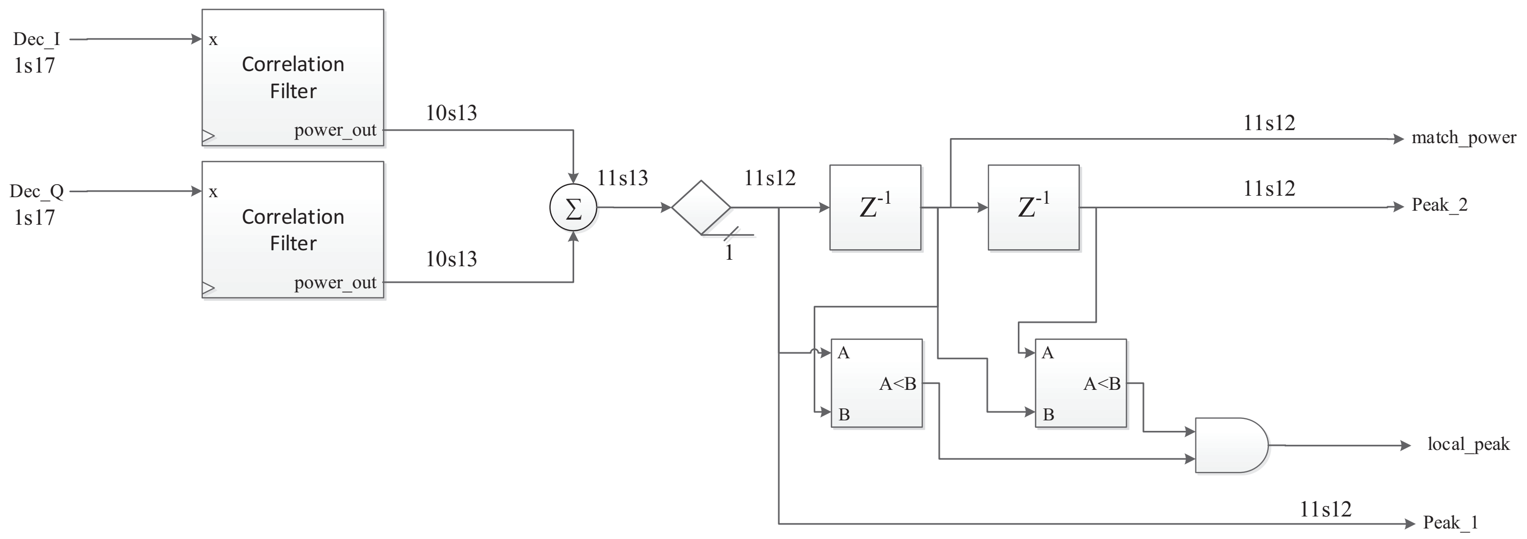


Figure 6.30: Schematic of the Timing Detector Circuit

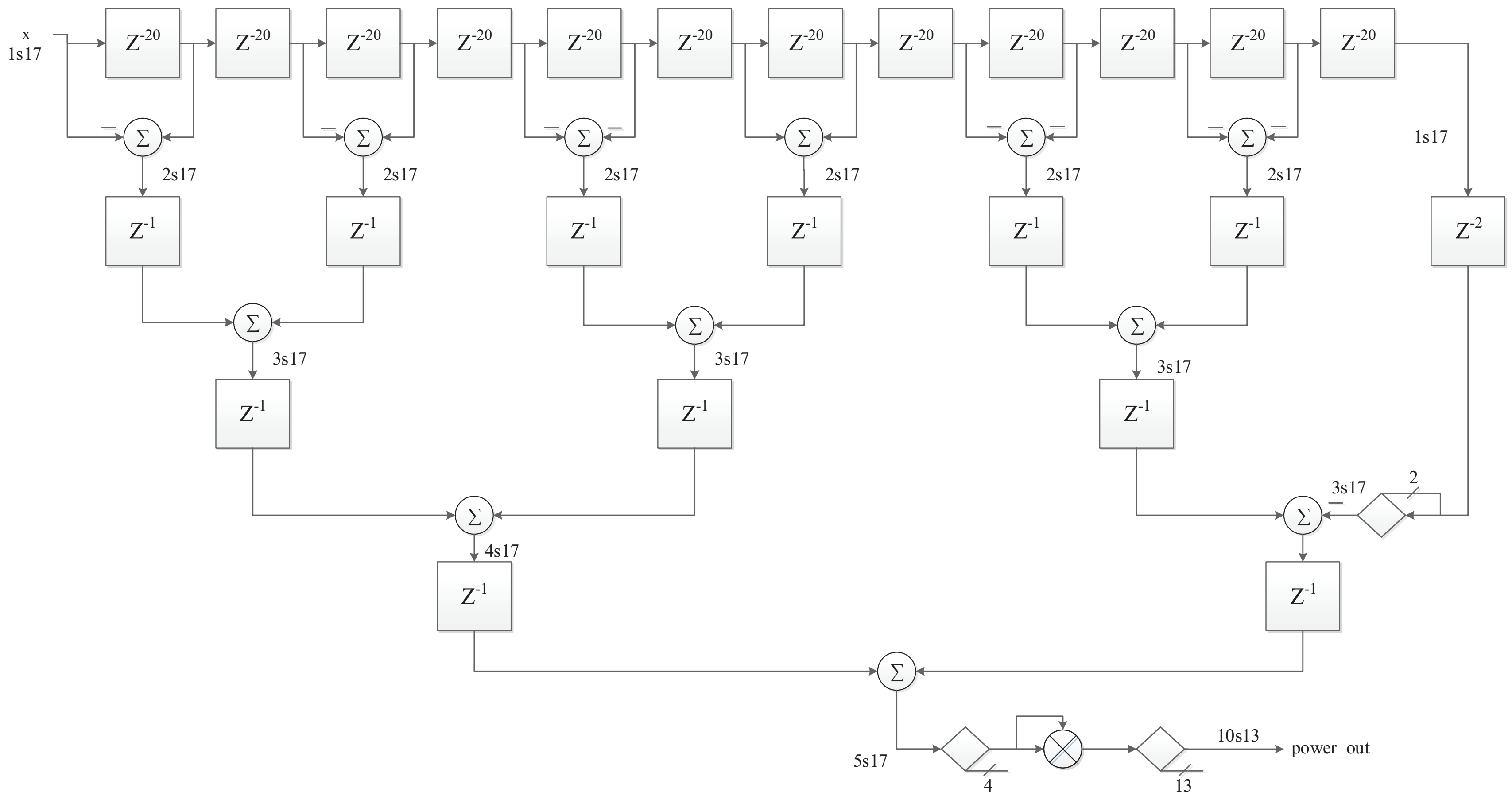


Figure 6.31: Schematic of the Correlation Filter Circuit

6.7.1 Testing the Timing Recovery Circuit

The timing recovery circuit was implemented in verilog HDL and tested by inserting sample delays between the modulator and demodulator and examining the MER of the demodulated signal signal. The sample delays inserted were varied between one and twenty to test all sample delay cases. The measured MER of the test cases was found to be 54.6 dB, 0.1 dB below the simulated value of MER.

Once the sample delay cases were tested, the matched filter coefficient sets were varied to test fractional delay detection. In order to perform this test, the feedback signal from the timing recovery circuit to the matched filter that selects the proper filter coefficient set was disconnected so that the coefficient sets could be varied for testing purposes. The value of the mf_adjust signal was then monitored in Signal Tap and compared to the selected coefficient set in the matched filter. In the tests performed if was found that the mf_adjust signal agreed with the selected coefficient set number.

6.8 Frequency Recovery

The implementation of the frequency recovery circuit described in section 5.2.5 is shown in the schematic in figure 6.32.

The CORDIC arctan circuits created for use in the frequency and phase recovery circuits produce an 18 bit result in a 1s17 format that represents [-π, π) radians that have been normalized. The two arctan values are then subtracted from each other to determine the rate of change in phase between the signals which can result in a 2s17 number. This value is then passed through the limiter circuit described in section 5.2.5. Twenty-six symbols, the equivalent of two preamble sequences, are then accumulated and averaged. The input to the accumulator must be sign extended to create a 5s17 formatted signal to accommodate the sum of the twenty-six symbols. The averaging is performed in two stages, a truncation causing a pseudo bit shift implementing a division by sixteen and a multiplication by a 0s18 number representing $\frac{16}{26(13)}$ implementing equation (5.9). The multiplication produces a 1s35 signal that is then truncated to a 27 bit number by removing the lower 9 bits and fed into two CORDIC based NCOs that despin the I and Q inputs of the circuit.

The outputs of the CORDIC NCOs are then summed to implement a multiplication of the I and Q phases by $e^{-j\omega}$ as given in (6.17), where ω is the frequency fed into the CORDIC NCOs. The results of the summations produce two 2s17 formatted signals which are then reduced to 1s17 signals formats by another pseudo bit shift.

$$(I + jQ)e^{-j\omega} = (Icos(\omega) + Qsin(\omega)) + j(Qcos(\omega) - Isin(\omega)) \tag{6.17}$$

6.8.1 Testing the Frequency Recovery Circuit

The frequency recovery circuit was tested by adjusting the frequency offset between the modulator and demodulator to twelve different values and 2048 tests at each frequency offset were taken with no AWGN added to the channel. The average error between the actual frequency offset and the frequency offset determined by the frequency recovery circuit was found to be less than 3 Hz and the variance found to be about 4.75x10^{-10} cycles^2/sample^2.

Another test was then run using the same twelve frequencies and another 2048 samples at each frequency with 25 dBc AWGN added to the channel. The AWGN was added using an open source AWGN generation circuit[1]. To achieve the 25 dBc of required AWGN, the signal power in the passband of the transmitted signal was measured using a signal analyzer. The signal was then disabled and the AWGN noise power was measured. The gain of the transmitted signal was then adjusted to produce the desired 25 dBc AWGN. Once again the average error between the actual offset frequency and determined offset frequency was found to be less than 3 Hz with a variance of 5.98x10^{-10} cycles^2/sample^2.

Results were obtained using Signal Tap from setting the frequency offset to 5 kHz and capturing 200 symbols of data immediately after the completion of the frequency recovery circuit and plotted in figure 6.33 along with the incoming I and Q data in the absence of AWGN. The plot clearly shows that the frequency recovery circuit is capable of removing most of the frequency offset from the incoming signal.

[1]The Gaussian Noise Generator module is available from https://opencores.org/project,gng.

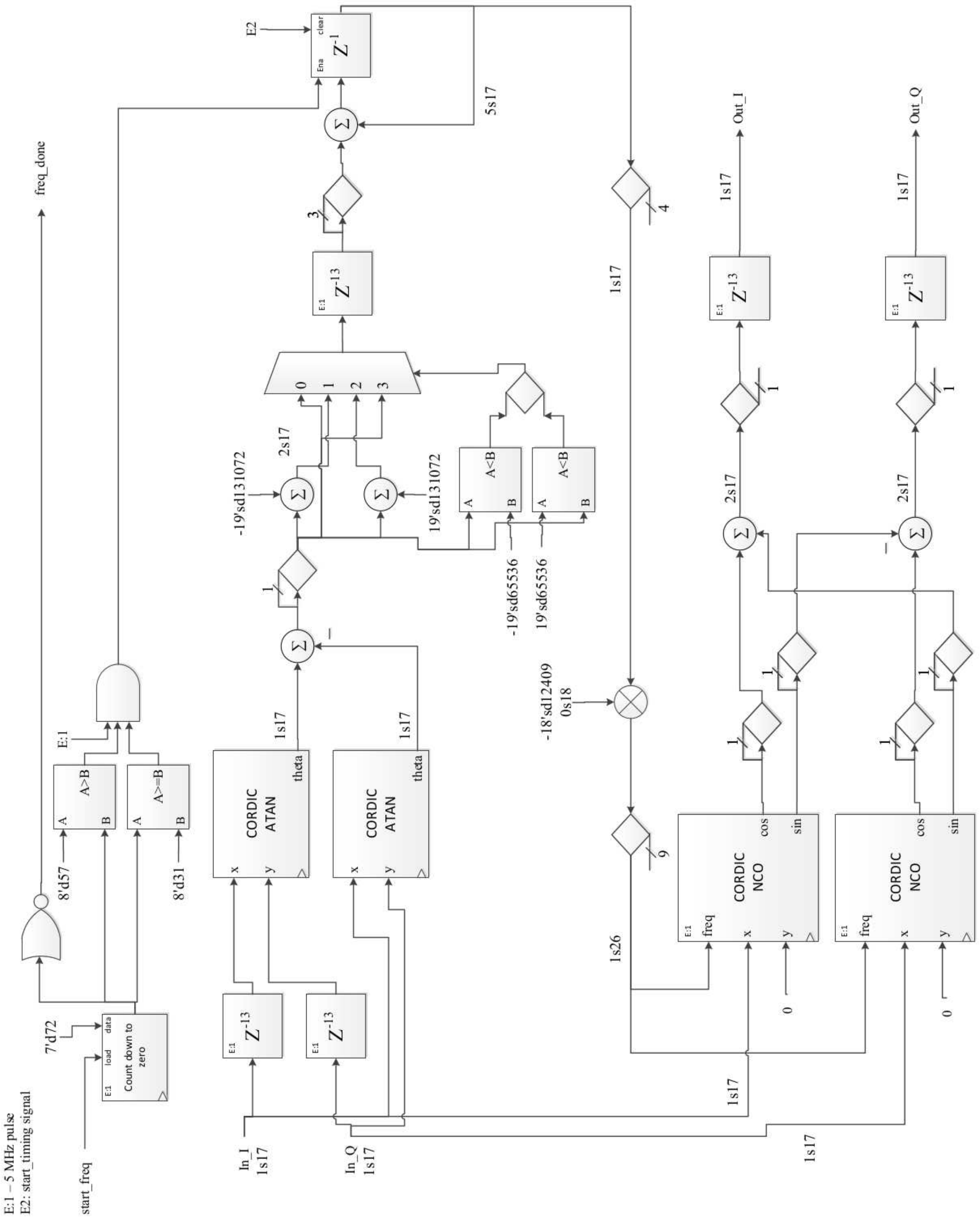


Figure 6.32: Schematic of Frequency Recovery Circuit

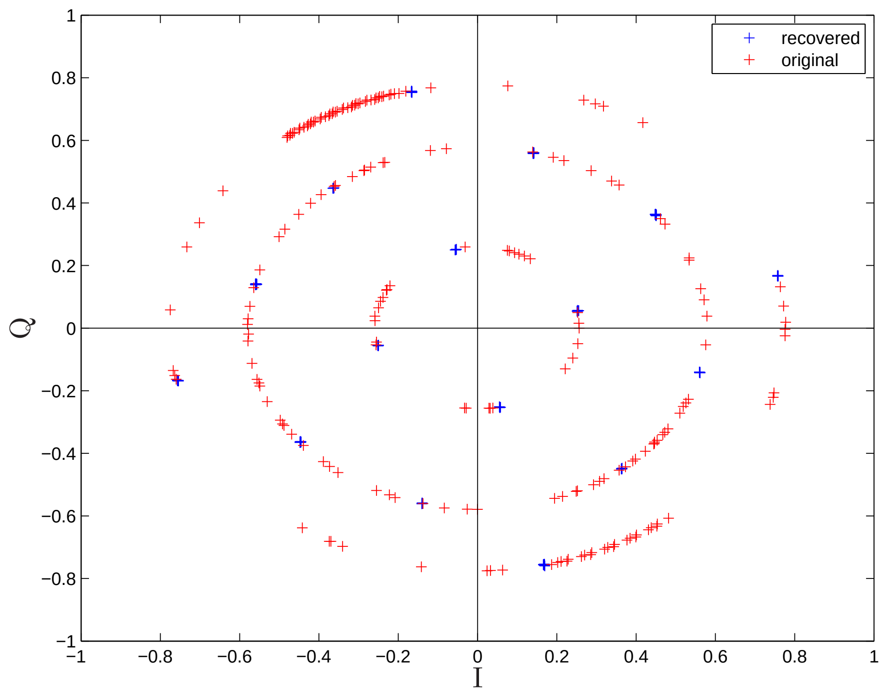


Figure 6.33: Constellation Plot of Recieved and Frequencey Corrected I and Q Data

6.9 Phase Recovery

The implementation of the phase recovery circuit described in section 5.2.6 is shown in the schematic in figure 6.34.

To remove the phase modulation the input signal is to be multiplied by the complex conjugate of the preamble sequence. In order to remove the multipliers used to perform the complex multiplication needed for phase demodulation, multiplexers were used. Since the preamble sequence is made up of QPSK symbols the complex conjugate of the symbols are simply the negative of the symbol. The multiplexers can produce the appropriate values by selecting the negative or positive of the input symbol depending upon the corresponding preamble sequence symbol. To align the input symbol to its corresponding preamble symbol the seq_start signal from the timing recovery circuit is used.

The output of the phase demodulator circuit creates a 2s17 signal format due to the adders used to implement the complex multiplication. The output of the adders is registered for two reasons: for pipelining purposes and to align the value of the down-counter with that of the accumulator circuit. A down-counter is used to keep track of the number of symbols accumulated and when its load signal is asserted it takes one symbol time for the loaded value to reach the output of the down counter, hence the need for the register. While it is possible to accumulate only one preamble sequence to determine the phase offset a preamble of 13 symbols this does not best utilize a binary signal format since it would have a gain of 13. Instead, at the expense of transmitting an extra preamble sequence, 16 symbols were accumulated to better utilize the 6s17 signal format created when designing the accumulator circuit. The accumulated value is then reduced to a 1s17 format through truncation of its LSBs and fed into a CORDIC arctan circuit. The output of the CORDIC arctan is then fed into a CORDIC rotator that corrects the phase offset of the symbols input to the phase recovery circuit and passes these values as a 1s17 format to subsequent circuitry in the demodulator for further processing.

6.9.1 Testing the Phase Recovery Circuit

Testing of the phase recovery circuit was done by adjusting the phase offset of the output of the CORDIC downconverter by inserting a CORDIC rotation circuit in the demodulator. The frequency recovery circuit was disabled and the frequency offset between the modulator and demodulator set to zero. The MER of the demodulated symbols after phase recovery were then captured using signal tap and imported into excel for analysis.

The first set of tests was conducted using seven randomly selected phase offsets without any AWGN inserted into the channel. 256 data points for each phase offset were captured. The analysis of all 1,792 points showed the average MER to be 54.3 with a variance of 0.03.

The second set of tests was conducted using seven randomly selected phase offset with 25 dB of AWGN inserted into the channel. 256 data points for each phase offset were captured. The analysis of all 1,792 points showed the average MER to be 30.6 dB with a variance of 1.26.

I and Q data was captured at the input to the circuit and the output of the circuit immediately after the phase done signal was asserted. The data collected were plotted to verify that the phase correction circuit worked correctly. One capture of the data is given in figure 6.35, which plots the corrected phase output, input signal, and the theoretical constellation points for the QAM signal.

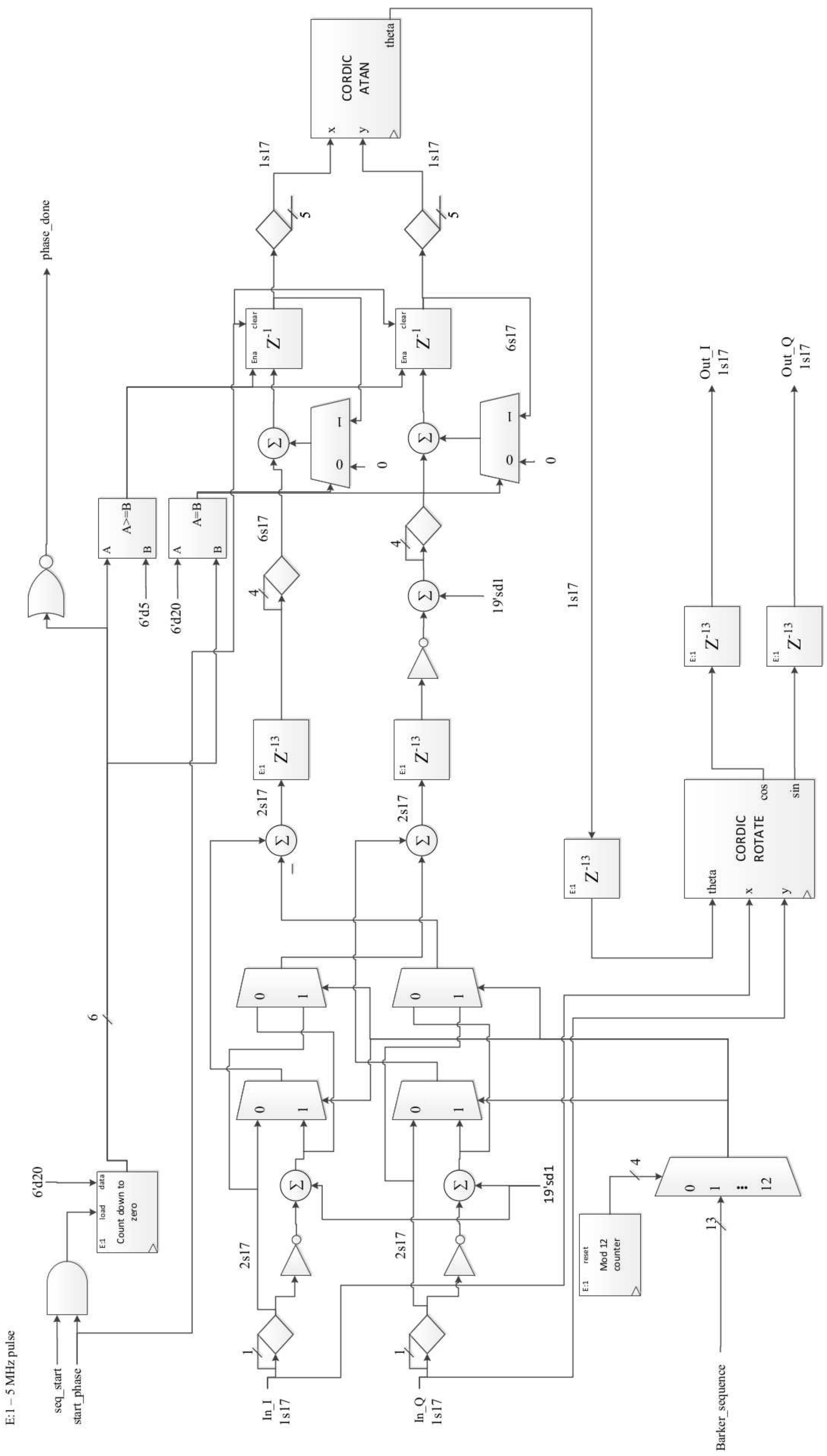


Figure 6.34: Schematic of Phase Recovery Circuit

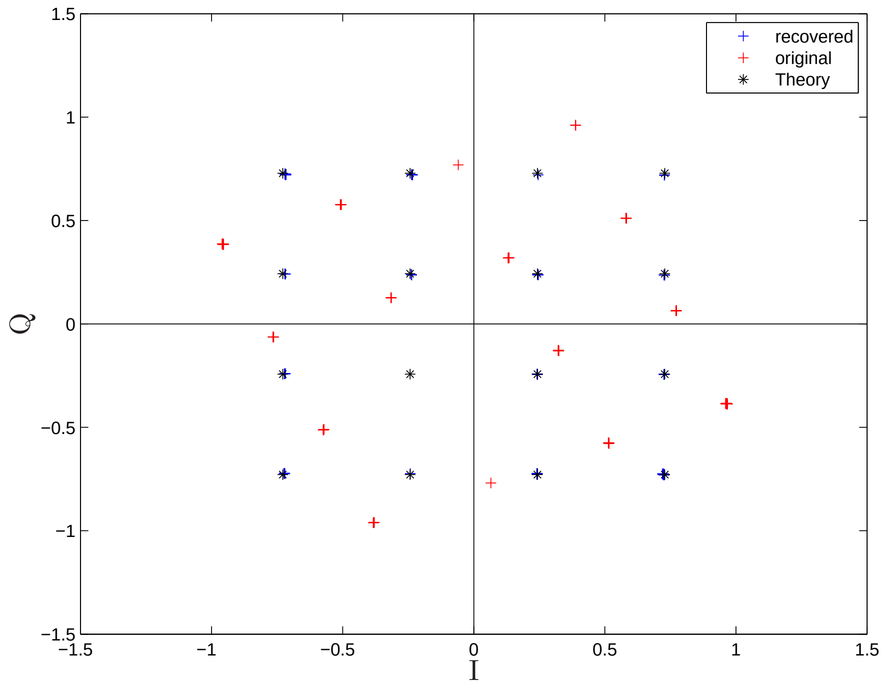


Figure 6.35: Constellation Plot of Recieved and Phase Corrected I and Q Data

6.10 Bit Error Rate Performance

Another test of the performance of the digital demodulator is to check the bit error rate (BER). The BER is a ratio of the number of received bit errors over the number of transmitted bits. This ratio gives an approximation to the probability of a bit error occurring and can be predicted theoretically.

For a grey-coded 16-QAM modulated signal that is passed through a AWGN channel, the theoretical BER of the demodulator can be given by (6.18)[25],

$$BER_{16\_QAM} = \frac{3}{8}\left(\text{erfc}\left(\sqrt{\frac{4E_b}{10N_o}}\right)\right), \tag{6.18}$$

where $\frac{E_b}{N_o}$ is the ratio of the average energy per bit and the noise energy in the channel.

The LFSR used to generate the transmitted symbols was synchronized to an LFSR at the demodulator, using the circuit given in appendix A.2, so that the transmitted symbols were available at the demodulator and compared to each other to determine if a received bit was in error or not. Once the LFSRs were synchronized the amount of AWGN in the channel was varied and error rates captured using Signal Tap in the Quartus software package over a period of 1,048,576 transmitted symbols. The captured values were then plotted against the theoretical BER rates as shown in figure 6.36. The plots indicate that the designed digital front end in the demodulator preforms as expected.

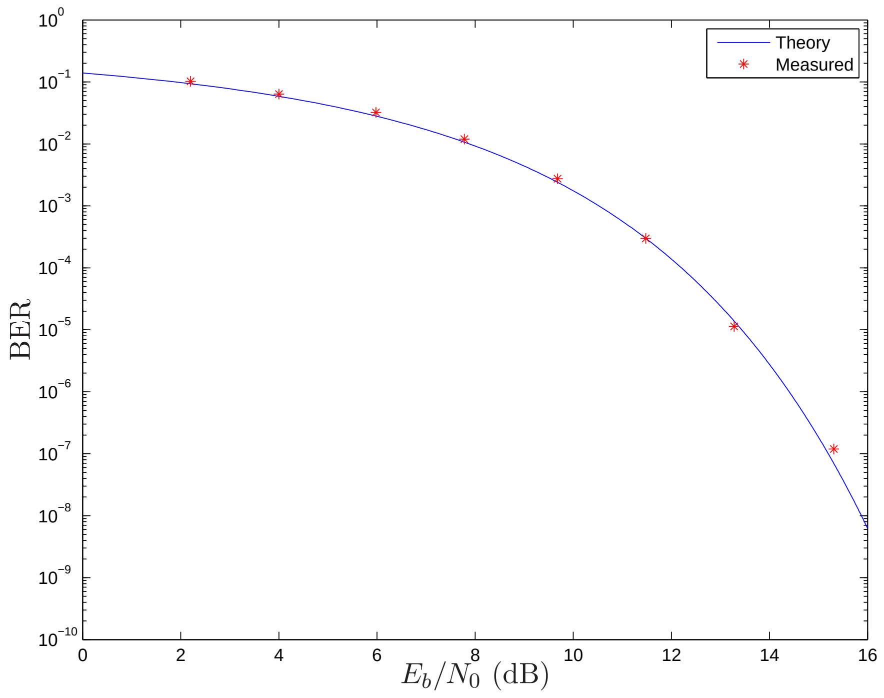


Figure 6.36: BER vs. $\frac{E_b}{N_o}$

Chapter 7

Conclusion and Future Work

7.1 Summary

The design and implementation of a DOCSIS 3.0 based receiver is a complex and time consuming task. Design choices and methods are nearly endless and each method has benefits and drawbacks. The specifications in DOCSIS 3.0 provide some constraints over the design choices made, as do hardware specifications. In addition, the receiver must also overcome common impairments found during data transmission through a channel.

The task of implementing a DOCSIS system, in a hardware efficient way, is challenging. The most costly hardware resource found in FPGAs are multipliers, thus in this thesis a focus on the reduction in use of multipliers was a priority. Although certain aspects of reducing multiplier usage may seem straight forward, the actual implementation of these methods can be fraught with unforeseen issues leading to significant increases in time spent debugging circuits.

This thesis aims to provide readers with design methods that can be used to cost effectively implement a receiver front-end on an FPGA that will perform according to the DOCSIS 3.0 specifications in an AWGN channel. In doing so, this thesis explains the design process used for hardware implementation and hardware reduction using no proprietary logic cores.

Performance analysis of the implementation of the digital front-end is done using simulation in MATLAB and data captured from the practical implementation of the circuits on a DE4 FPGA development board.

7.2 Contributions

The primary contributions of this work are methods that can be used to implement multiplier efficient circuitry for the front-end of a DOCSIS 3.0 compliant receiver. These methods are then used to perform an example implementation as a guide for researchers and designers who want to implement the digital front-end of a DOCSIS 3.0 compliant receiver.

In producing the circuits required, this work also ensures that no proprietary logic cores are used in the digital front-end of the receiver. The elimination of proprietary cores gives designers control over their circuit implementations.

A guided design of bit path sizing of the implemented circuitry, including proper signal sizing for filters and other circuits, as well as some discussion on the effects of moving from infinite precision to finite precision systems is discussed. This work also provides receiver designers with a design methodology that can speed up development time by reducing learning time, since few resources that cover the entire implementation of a DOCSIS 3.0 compliant receiver front-end exist that do not use proprietary logic cores.

7.3 Resource Usage and Overall Performance

Since resources used in designs implemented by CATV equipment manufacturers are closely guarded and hardware constraints play a role in the implementation of each front-end design, a direct resource usage comparison between the example design presented here and other existing solutions is not possible. Instead, table 7.1 is given, for comparative purposes, to show the number of multipliers used to implement the circuits in the design example and the number of multipliers that would be required to implement the same circuits if no reduction techniques such as time-sharing, multiply and accumulate and successive approximation circuits were used.

Table 7.1 also provides the total number of multipliers used to implement a single phase and the number used to implement both phases (both the in-phase and the quadrature-phase paths) of a QAM receiver front-end. When both phases are considered, some circuitry must be duplicated, such as the matched and anti-aliasing filters. Overall the number of multi-

Table 7.1: Non-reduced and Reduced Multiplier Usage by Circuit

Circuit	**Non-reduced**	**Reduced**
Downconversion	2	0
Antialiasing (per phase)	15	8
Matched filter (per phase)	33	7
Timing Recovery	60	34
Frequency Recovery	8	1
Phase Recovery	8	0
Total (per phase)	126	50
Total (both phases)	174	65

pliers required for implementation can be reduced by greater than 60% with the reduction techniques discussed in this work.

The total resource usage of the DE4 FPGA, with all auxiliary circuits used, including those discussed in the appendices, as well as the AWGN generation circuit and the modulator used for testing, was: 11% logic utilization, <1% memory bits (23,753/21,233,664 bits), and 288/1,024 18-bit multipliers. These resource usage figures are provided here for comparative purposes for those who have previously developed similar hardware. It should be noted that the majority of the multipliers used were in the MER measurement circuit, which was used for testing purposes.

The performance testing of the implemented demodulator front-end achieved a MER of 54.3 dB without AWGN and 30.6 dB with 25 dBc AWGN. The simulated theoretical values for the MER of the system were determined to be 54.7 dB without noise and close to 31 dB with 25 dB AWGN. These results indicate that, if the design methods presented here are followed, a DOCSIS 3.0 compliant receiver can be successfully implemented with an implementation loss of 0.4 dB.

7.4 Future Work

The work presented in this thesis considers only an AWGN channel; in practice it is highly likely that there will be additional transmission channels located next to the desired channel being demodulated. If these adjacent channels are present it would beneficial to be able to further adjust the matched filter coefficients used in the demodulator to improve the MER of the demodulator as presented in [12]. The implementation of circuitry that could measure the power in adjacent channels relative to the power in the channel of interest could be used to adjust the matched filter coefficients thereby improving overall MER of the demodulator when adjacent channels are present.

Some further reduction of resources could be performed on the circuitry presented in this work. This work implements the design of the demodulator in a modular fashion creating blocks that can be used to implement the filters for each of the in-phase and quadrature phase components of the received signals at the demodulator. For further resource efficiency these designs could be combined so that both the in-phase and quadrature phase portions are contained in the same modules. In doing so it becomes possible to use left over multiplication cycles in circuits that have time-shared multipliers.

In order to determine the overall performance of a receiver implemented using the methods presented here, an equalizer should be implemented and added to the design to reduce possible frequency offsets remaining after the frequency correction circuitry. Once completed the overall system performance could be measured and examined.

References

[1] CableLabs certified and qualified devices. http://www.cablelabs.com/specs/certification/. Accessed: 2017-06-15.

[2] CableLabs member companies. http://www.cablelabs.com/about-cablelabs/member-companies/. Accessed: 2017-06-15.

[3] *Data Conversion HSMC Refernce Manual*.

[4] MAX5880 high-density downstream cable qam modulator and digital upconverter. https://www.maximintegrated.com/en/products/comms/wireless-rf/MAX5880.html. Accessed: 2018-02-16.

[5] B. Berscheid, E. Salt, and H. H. Nguyen. An economical, isi-immune frequency offset estimator for docsis upstream channels. *IEEE Transactions on Broadcasting*, 58(2):310–316, June 2012.

[6] Brian Berscheid. *FPGA-Based DOCSIS Upstream Demodulator*. PhD thesis, University of Saskatchewan, 2011.

[7] Cable Television Laboratories, Inc. *Data-Over cable Service Interface Specifications DOCSIS 3.0: Physical Layer Specification*, cm-sp-phyv3.0-i13-170111 edition, January 2017.

[8] Leon W. Couch. *Digital and Analog Communication Systems*. Prentice Hall, 6th edition, 2001.

[9] M. S. Devi, M. Saketh, and Muthumeenakshi. Design of costas loop for carrier recovery mechanism of bpsk demodulation. In *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 424–429, June 2017.

[10] David Fellows and Doug Jones. DOCSIS^{TM}cable modem technology. *IEEE Communications Magazine*, pages 202–209, March 2001.

[11] F. Gardner. A bpsk/qpsk timing-error detector for sampled receivers. *IEEE Transactions on Communications*, 34(5):423–429, May 1986.

[12] R. Gowen, B. Daku, and D. Dodds. Optimizing a matched filter in the presence of isi and adjacent channel interference. In *2013 26th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–4, May 2013.

[13] Fred Harris and Dragan Vultec. Multirate signal processing for communication systems - multirate_script:lineardesign.m. http://www.informit.com/store/multirate-signal-processing-for-communication-systems-9780137009053. Accessed: 2018-02-17.

[14] Fredric J. Harris. *Multirate Signal Processing For Communication Systems*. Prentice Hall, 2004.

[15] Simon Haykin. *Communication Systems*. John Wiley and Sons, Inc., 4th edition, 2001.

[16] Ronald K. Jurgen. Two-way applications for cable television systems in the '70s. *IEEE Spectrum*, pages 39–54, November 1971.

[17] S. Kalle, M. R. Choudhury, F. Bui, and D. E. Dodds. Performance of unique word timing offset estimator. In *2013 26th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–4, May 2013.

[18] Joseph R. Kiniry and Christopher Metz. CABLE MODEMS: cable TV delivers the internet. *IEEE Internet Computing*, pages 12–15, May-June 1998.

[19] A. Krukowski, R. C. S. Morling, and I. Kale. Quantization effects in the polyphase n-path iir structure. *IEEE Transactions on Instrumentation and Measurement*, 51(6):1271–1278, Dec 2002.

[20] Ian Kuon and Jonathan Rose. Measuring the gap between FPGAs and ASICs. *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, 26(2):203–215, February 2007.

[21] K. Narendra, B. J. Shabir Ahmed, K. Swaroop Kumar, and G.H. Asha. FPGA implementation of fixed point integer divider using iterative array structure. *IJETR*, 3(4):170–179, April 2015.

[22] Alan V. Oppenheim, Ronald W. Schafer, and John R. Buck. *Discrete-Time Signal Processing*. Prentice Hall, 2nd edition, 1998.

[23] Chris Pearson. *High Speed, Digital to Analog Converters Basics*. Texas Instruments, October 2012.

[24] Stephen Perkins and Alan Gatherer. Two-way broadband CATV-HFC networks: state-of-the-art and future trends. *Computer Networks*, 31:313–326, 1999.

[25] Krishna Sankar Madhavan Pillai. Symbol Error Rate (SER) for 16-QAM. http://www.dsplog.com/2007/12/09/symbol-error-rate-for-16-qam. Accessed: 2017-08-08.

[26] S. O. Popescu, A. S. Gontean, and D. Ianchis. Implementation of a qpsk system on fpga. In *2011 IEEE 9th International Symposium on Intelligent Systems and Informatics*, pages 365–370, Sept 2011.

[27] Michael Rice. *Digital Communications A Discrete-Time Approach*. Pearson Pentice Hall, 2009.

[28] J. Eric Salt and Ha H. Nquyen. EE 811 - digital signal processing for communications - lecture notes. University of Saskatchewan, October 2011.

[29] E. Strathford Smith. The emergence of CATV: A look at the evolution of a revolution. In *Proceedings of the IEEE*, volume 58, pages 967–982. IEEE, July 1970.

[30] STMicroelectronics. *STV0297J QAM demodulator IC with A/D converter - Data Brief*, December 2006.

[31] Vlatko Čerić. Advancements and trends in the world wide web search. In *ITI 2000. Proceedings of the 22nd International Conference on Information Technology Interfaces (Cat. No.00EX411)*, pages 211–220, June 2000.

[32] J. E. Volder. The CORDIC trigonometric computing technique. *IRE Transactions on Electronic Computers*, EC-8(3):330–334, Sept 1959.

Appendix
Additional Circuits

A.1 Integer Division

The integer division circuit implemented in this thesis is based upon the restoring array divider presented in [21]. The practical implementation of the circuit is given in figures 1 and 2.

The circuit has been pipelined and broken down into stages. The circuit operates on a single bit of the dividend in each stage of the divider circuit. Throughout each stage a single bit of the dividend is appended to the remainder of the previous stage and fed into the next stage as the dividend. The full explanation of the algorithm is presented in [21] and will not be gone into great detail here. The output of each divider stage produces a single bit of the quotient and a remainder that is of the same length as the dividend.

The implementation used here is such that the dividend (labeled "numer", in the circuit diagram) and divisor (labeled "denom", in the circuit diagram) are of the same length, in this case 23 bits. Since the timing recovery circuit that uses this divider can produce both positive and negative numbers, the implementation presented in [21] must be modified to implement a signed integer division. To achieve the ability to implement the signed division the first MSB of the dividend and divisor are examined by using an exclusive-or operation on the bits. If the result of the exclusive-or is one then the resulting quotient must be negated, otherwise the quotient remains unchanged. It should also be noted that if the dividend or divisor are negative valued they must have a two's complement taken prior to performing the integer division operation.

The circuit presented in figure 2 produces a quotient that is 23 bits in length, which is not the same as the circuit used in the design but is presented this way here for ease of understanding. The actual implementation, while still using 23 bits for the remainder, divisor and dividend, retains only the last 6 bits of the quotient output of the divider circuit. Only the final 6 bits are required in this implementation since the values that the divider produce may only be ±32 as explained in the timing recovery section of this thesis.

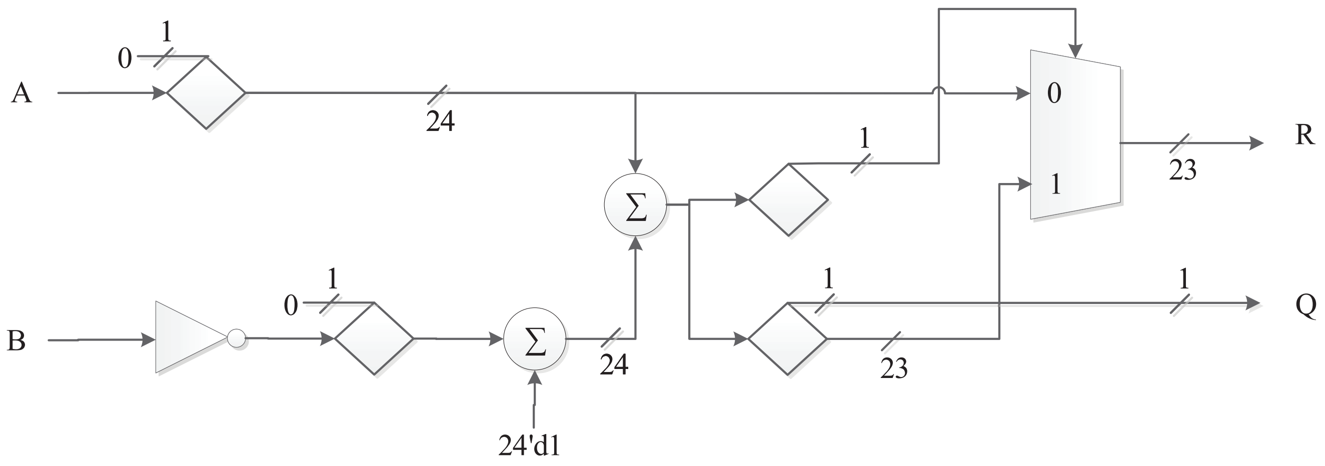


Figure 1: Divide Unit Circuit

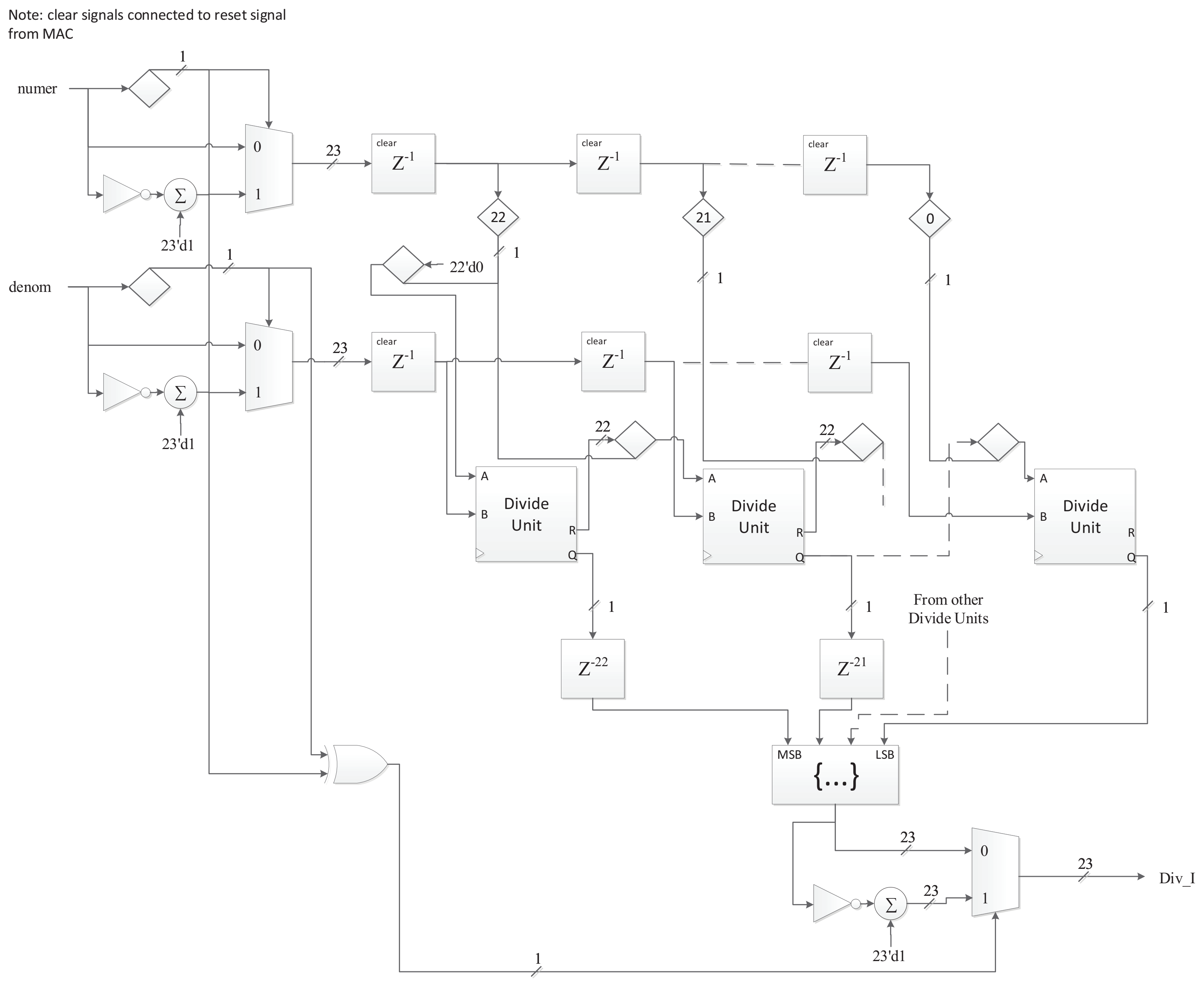


Figure 2: Signed Integer Division Circuit

A.2 Bit Error Rate Circuit

The bit error rate circuit used for testing the digital front end is provided in figure 3. The circuit consists of a 22-bit LFSR that is matched to the 22-bit LFSR that is used to generate the transmitted symbols in the modulator. The LFSR used in both cases uses feedback taps 7, 20, 21, and 22 to provide a maximum length pseudo-random sequence of length $2^{22} - 1$ values. In order to synchronize the transmitter and BER LFSRs a BER_start signal is issued. This signal resets the LFSR in the BER circuit to be all ones. The received symbols are then fed into the LFSR at N times the symbol rate of the LFSR, where N is the number of bits in the received I and Q phases of the QAM signal (in the 16-QAM transmitted signal used here N is equal to four). Once the first zero reaches the output of the LFSR an initialize flag is cleared, that is initially set with the BER_start signal. When this flag is cleared the LFSR output is then fed back into the LFSR input instead of using the received symbols.

After the two LFSRs have been synchronized with the same data, the received data is compared with that of the BER circuits LFSR input. If the two values differ an accumulator that keeps track of the total bit errors is incremented by one, otherwise it remains the same. After a period of $2^{22} - 1$ bits the accumulator is reset and begins accumulating values again. When the accumulator is reset the value accumulated just prior to the reset is taken and divided by N times the number of symbols accumulated over to determine the bit error rate of the system, where N is the number of bits per symbol.

It is important to note that when synchronizing the LFSRs in the transmitter and BER circuit that there must be no errors received otherwise the LFSRs will not generate the same values resulting in higher bit error rates than expected. To accomplish this task any noise sources must be disabled while the LFSRs are synchronized.

E1 – bit_rate_clk_ena pulse, 4 times the symbol clock rate

Note: set signals of LFSR connected to start BER signal from MAC

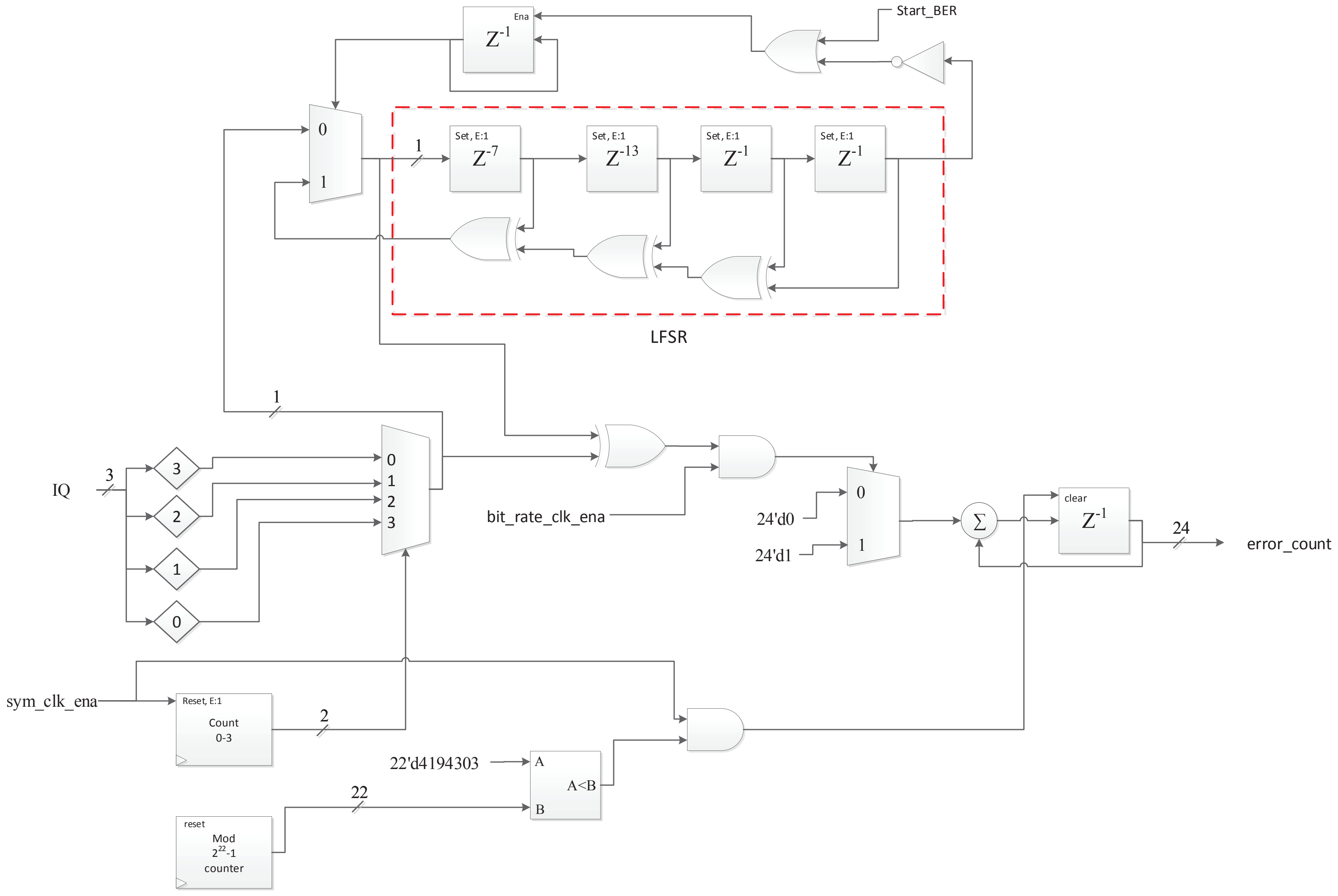


Figure 3: Bit Error Rate Circuit

A.3 MER Circuits

In order to determine the MER of the overall system and its components a MER measurement circuit was created to perform the MER calculation. The circuit used to determine the MER is shown in the schematic in figure 4. In order to speed up the development time of the circuit and the fact that the circuit is not a necessary part of the digital front end, the circuit was created using proprietary IP core blocks provided in the Quartus software suite.

The circuit uses the received I and Q signals after passing through all the recovery stages to determine values needed to complete the computation of the MER. In order to determine these reference values a second circuit is used. This secondary circuit was called the reference finder circuit and its schematic is given in figure 5.

The reference finder circuit is duplicated for both the I and Q received data (the `dec_var` signal shown in the circuit schematic). The circuit uses the dec_var value to determine the average value of the the received I and Q signals which in turn is used to determine the ideal received symbols' value. The ideal symbols' values are determined by placing the received I and Q data through a slicer circuit that determines the closest symbol to the received I and Q data. These symbols are then fed back into a mapper circuit along with the average value of the received I and Q data in order to determine the ideal values of the symbols. This is done in order to remove possible DC offsets from the received data since the values generated from the mapper will contain no DC. This mapper output is used to compute the average power in the ideal mapped values (avg_power_dec) as well as to produce an error signal.

The error signal is produced by subtracting the output of the mapper circuit from the decision variable (dec_var) in order to determine the error from the ideal received signal values. The error signal is then used to compute the average error (avg_error) and average squared error (avg_sq_error) over 2^{22} symbols.

The signals created in the reference finder circuits are then fed into the MER circuit. To retain precision of the values from the reference finder circuit the values are converted from fixed point to floating point using a proprietary IP core. The core produces 32 bit floating point values for each of the input signals to the MER circuit. These signals are then used to compute the MER as given by (1).

The set of IP cores provided does not include a log_{10} function so the circuit implements this functionality by using a natural log and division instead. The result of the calculation is then converted back from floating point to a fixed point representation in order to be viewed in Signal Tap. The fixed point representation chosen here was 7s11 format. This format allows for up to 64 dB of MER to be measured which exceeds the designed MER of the system.

$$MER = 10\log_{10}\left(\frac{(avg\_power\_dec\_I + avg\_power\_dec\_Q)}{(avg\_sq\_error\_I - avg\_error_I^2) + (avg\_sq\_error\_Q - avg\_error_Q^2)}\right) \tag{1}$$

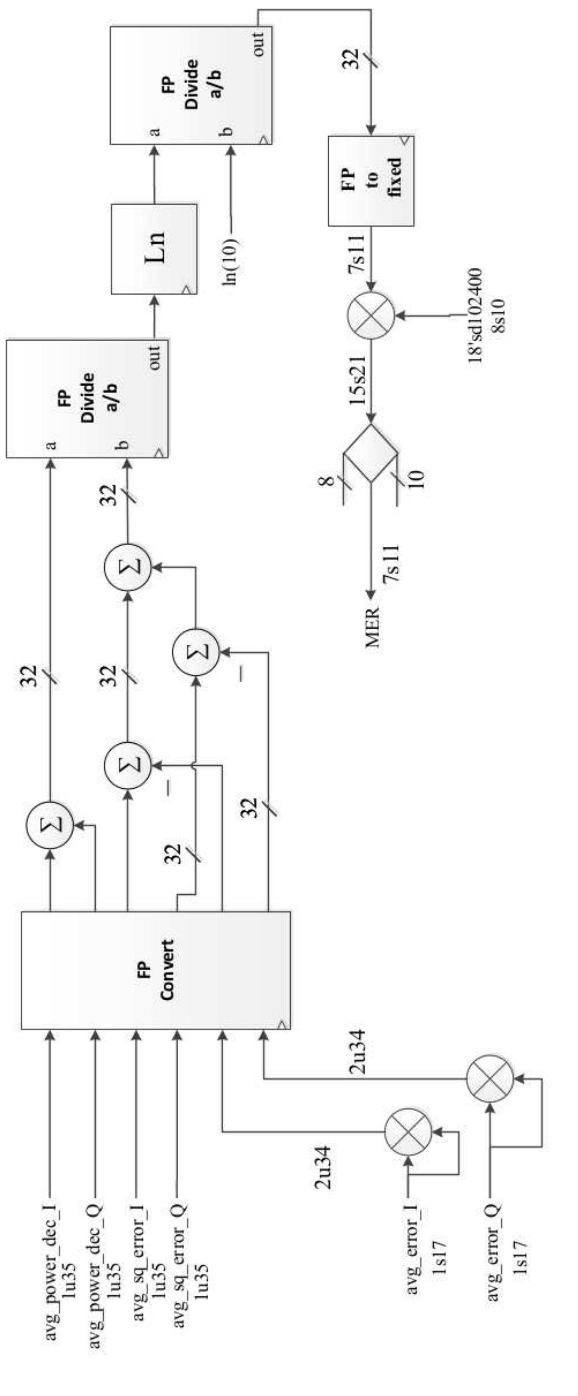


Figure 4: MER Calculation Circuit

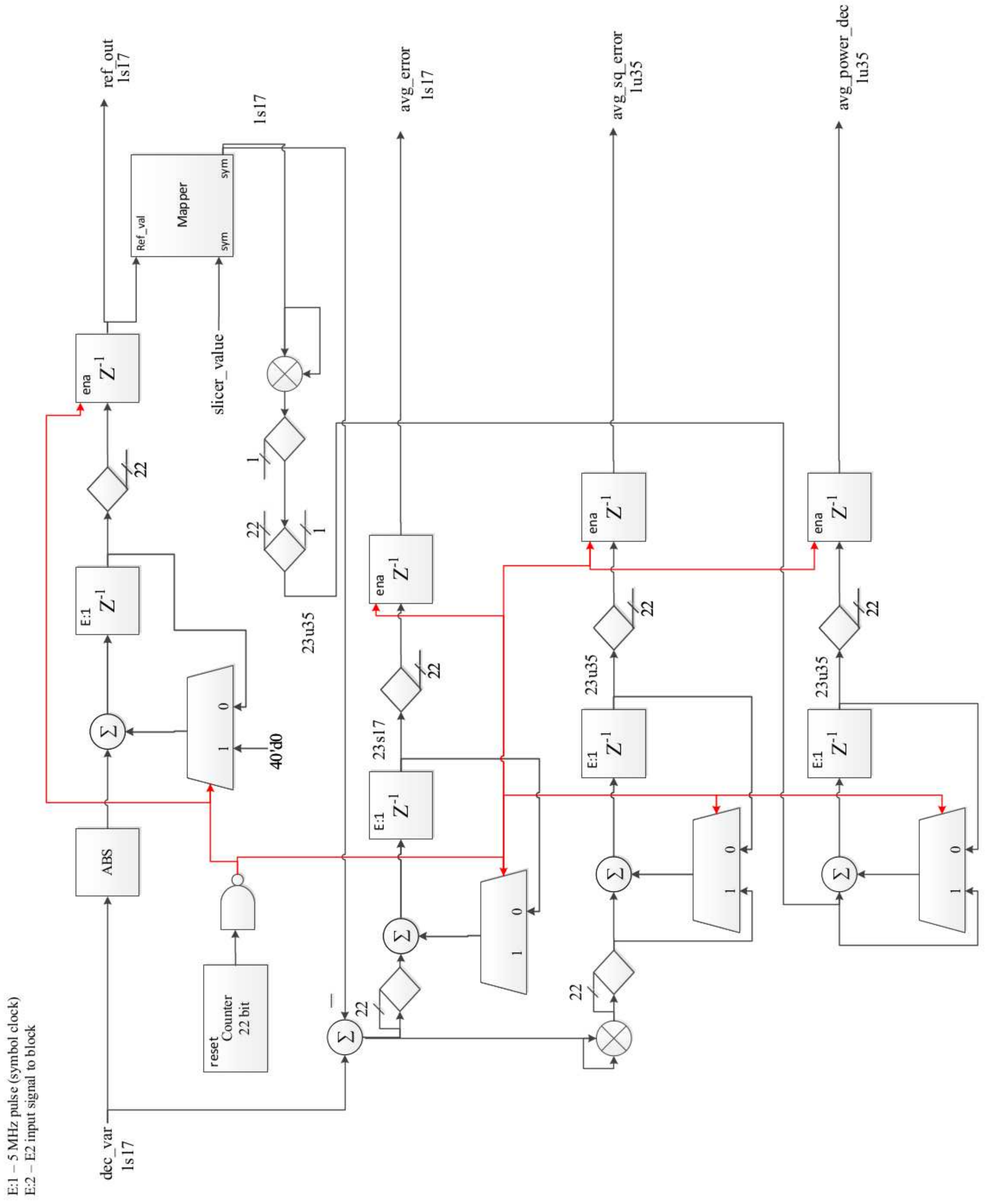


Figure 5: Reference Finder Circuit