AN ABSTRACT OF THE THESIS OF

Yichen Zhao for the degree of Master of Science in Electrical and Computer Engineering presented on May 28, 2013.

Title: Design and FPGA Implementation of Digital Transmission over Severe ISI Channels

Abstract approved:

Huaping Liu

Inter-symbol interference is one of the major factors that make the realization of high-data-rate digital communications system complex. Current designs face two main challenges: how to efficiently utilize the available bandwidth and how to reduce the hardware complexity of the transmitter and receiver. Traditional solutions use a single-band architecture. When ISI is severe, it might require an equalizer to mitigate ISI, which usually results in a high complexity and power consumption. This thesis focuses on the analysis and FPGA implementation of a multiband communication architecture. This design ensures that the channel frequency response in each sub-band is approximately flat to avoid the need of an equalizer. Specifically, a four-band

architecture is presented in detail, and this design is compared with the single-band approach.

First, basic theories are provided for convenience of understanding the major development in this thesis in terms of simulation and FPGA implementation. Then the channel characteristics, such as the frequency and impulse responses, are analyzed for a four-band architecture. The single- and four-band architectures are introduced separately and optimized in detail. The simulation results of both architectures are verified through FPGA implementation in the Xilinx Virtex5 development board. Finally, BERs of the two architectures are compared from both simulation and FPGA implementation results.

© Copyright by Yichen Zhao

May 28, 2013

All Rights Reserved

Design and FPGA Implementation of Digital Transmission over Severe ISI Channels

by

Yichen Zhao

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented May 28, 2013
Commencement June 2013

Master of Science thesis of Yichen Zhao presented on May 28, 2013

APPROVED:

Major Professor, representing Electrical and Computer Engineering

---

Director of the School of Electrical Engineering and Computer Science

---

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Yichen Zhao, Author

ACKNOWLEDGEMENTS

First at all, I express my deepest gratitude to my advisor, Dr. Huaping Liu, for his invaluable help both academically and personally during my three years of study in Corvallis. His broad knowledge, generous guidance and encouragement always give me confidence to overcome the difficulties in research. He is a role model for us to follow. I can't possibly finish my master thesis without his help.

I thank Dr. Alan Wang, Dr. Bechir Hamadoui and Dr. Chinweike Eseonu for their effort and time to serve on my thesis committee.

I would also express my thanks to my friends and group members, for their friendship and help. Special thanks go to Guanghua Shu, Tianzhu Qiao, Yao Liu, Zhenqiang Su and Arash Abbasi for their helpful discussion and advice.

Veronica and Ted have been like parents to me even since I came to Corvallis. They have helped me tremendously in every aspect of my life; when I do not have a shelter right after I came to Corvallis, it was you who gave me a home; when I encountered difficulties in studies, your encouragement gave me the strength. Forever I thank you for your kindness and unconditional love. Veronica, I will forever remember your words 'I appreciate your asking for help from us while you need help' and 'I am your America Mom'.

Finally, without the unconditional love and support from my parents Lixiong Zhao and Hongmei Yao, it is impossible for me to even have the chance to study in the USA. Thank you for giving me everything to prepare for my future and for forgiving me even when I make naive decisions. My deepest love and appreciation go to my girlfriend, Yizhe Yang, for her companionship, her encouragement, and her understanding and strong support during tough time periods. Because of you, my memory of life in this beautiful OSU campus will never fade.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (continued)

LIST OF FIGURES (continued)

Figure Page

LIST OF TABLES

Table Page

1. Introduction

1.1 Overview

This thesis focuses on simulation, FPGA implementation and specification of a multiband architecture for communication over severe frequency channel. The main advantages of the multiband architecture are its low hardware complexity since no equalizer is required and a lower bit error rate (BER) compared with the single-band architecture. This thesis builds on the work in [1]. It involves the allocation of sub-channels, the specification of filters used to minimize inter-symbol interference (ISI) and BER analysis. While the design can be applied for channels with any bandwidth, implementation will be over a telephone line, which is typically made of copper. A comprehensive description of the channel will be presented in detail, followed by digital signaling that employs a raised cosine filter in the transceiver, and FPGA implementation. Finally, advantages of the multiband design are presented and compared with the traditional single-band approach.

1.2 Background

When digital signaling speed requires a bandwidth that exceeds the typical operation bandwidth of a transmission media, the signal transmitted might be distorted by ISI. The channel between the transceivers at both ends of a line can be regarded as a low pass filter. The channel is optimized for transmitting signals with frequency up to the cutoff frequency. After the cutoff frequency, the attenuation gradually increases. For experiments in the lab,

we have to access to a short telephone line (to implement a practical channel, it requires line length in the range of 500m to 2km). With the telephone line in the lab, 3 dB attenuation is found at 54.25 KHz. At 84.09 KHz and 151.9 KHz, the attenuations are 21 dB and 130 dB, respectively. The maximum rate achieved with the four-band architecture is 217 Ksps. To prove the benefits of the multiband architecture, the conventional single-band architecture is implemented to transmit at the same bit rate. Binary modulation is adopted for the single-band architecture, which results in the maximum frequency of at least 108.5 KHz. Due to the high attenuation of over 100 dB at this frequency, the receiver requires an equalizer to mitigate ISI. The corresponding simulated BER for the single-band is 16.74% due to severe ISI without an equalizer.

An improved architecture is to divide the power spectrum into smaller bands, called sub-channels, where parallel data streams use these sub-bands. The channel is approximately frequency flat in each sub-band, which lowers the hardware complex since an equalizer in the receiver is not required. This design allows the use of a higher-order modulation in each subband, which compresses the overall bandwidth and makes the circuit design easier because of the lower symbol rate in each subband. For the specific example of the four-band architecture, it is an acceptable choice to apply QPSK and 16-QAM for the baseband and other three subbands. Within this scheme, the total data transmission requires the frequency spectrum from 0 to 84.09 KHz. The simulated BER for the four-band architecture is as low as 0.99%.

All Matlab simulation results have been verified through FPGA implementation. The components needed for this architecture such as the mixer, filter and pseudo-random code

generator, are all created by using Verilog and simulated by using Modelsim. The Xilinx ISE is capable of synthesizing and implementing a design based on Verilog codes to create a programming file. After downloading the file into FPGA, ChipScope acts as a virtual logic analyzer to detect the internal signal and to debug. The results from FPGA implementation match well Matlab simulation results, and both prove the efficiency of the four-band architecture.

1.3 Thesis Organization

The first chapter briefly describes the design goal, some background, and organization of this thesis. The second chapter covers most of the basic theories utilized in this thesis. Chapter 3 discusses the conventional baseband architecture and includes simulation results of the single-band. Chapter 4 describes the design of the four-band architecture which consists of filter design and band partitioning, and compares the four-band architecture with the single-band architecture. The final chapter presents the details of FPGA implementation, and compares the simulation results given in Chapter 4 and Chapter 5 with FPGA implementation results.

2. Theory Basics

2.1 Intersymbol Interference

Intersymbol interference arises in pulse-modulation systems whenever one transmitted pulse leaks into future transmission periods [2]. ISI is first detected in the message over the transatlantic telegraph cables. Fig. 2.1 shows how ISI errors when decision is made in the receiver. At the "ERRORED ZERO" in Fig. 2.1, the transmitted data is in sequence '1, 0, 1'. Due to ISI, the two adjacent 1's will distort '0' and make it '1'. The received signal of '1' is not the transmitted bit '0', and thus a decision error occurs. ISI is different from Gaussian noise; it originates from the signal, but noise is an additive distortion independent of the transmitted signal.

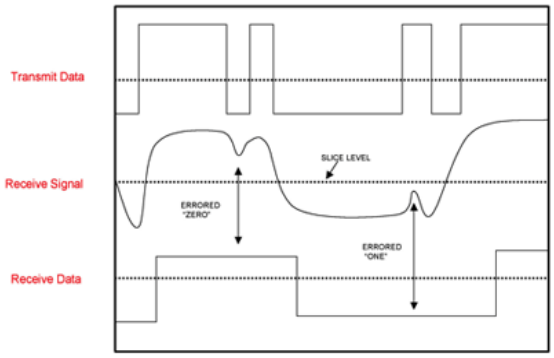

Figure 2.1: Errors caused by inter-symbol interference.

ISI influences the digital transmission over the channel, but what causes this unwanted phenomenon? One of the causes is multipath propagation. It commonly happens in wireless channels when the signal reaches the receiver through different paths. The cause of multipath propagation includes reflection, refraction and atmospheric effect.

Another cause is the band-limited channel. When the frequency components of the transmitted signal reaches over the cutoff frequency of the channel, different frequency components do not experience equal attenuation, which distorts the signal. As shown by the example in Fig. 2.1, the square pulse of the transmitted signal contains both very high and very low frequency components. The attenuation of high and low frequency components are different and the phase response is nonlinear, which results in the energy in high frequency components to spread out to subsequent symbols. The amplitude of the signal will shrink exponentially if its bandwidth exceeds the cutoff frequency of the channel. The usual solution to mitigate ISI is to reduce the bandwidth of the signal or to use an equalizer, which attempts to reverse the distortion to the transmitted signal.

2.2 Nyquist Criterion

To transmit data through a bandlimited channel without ISI, it is well known that the Nyquist criterion must be satisfied [3]. For digital communications, we only need to focus on eliminating ISI at sampling points only.

Let the channel frequency be $H(f)$. For a rectangular $H(f)$, that is

$$H(f) = \begin{cases} T & |f| \le 1/2T \\ 0 & Otherwise \end{cases} \tag{2.1}$$

its inverse Fourier transform is

$$h(t) = \begin{cases} \dfrac{\sin(\pi t/T)}{(\pi t/T)} = \sin c(t/T). \end{cases} \tag{2.2}$$

The Nyquist criterion is:

$$h(\mathrm{n,T}) = \begin{cases} 1; & n = 0 \\ 0; & n = 1 \end{cases} \tag{2.3}$$

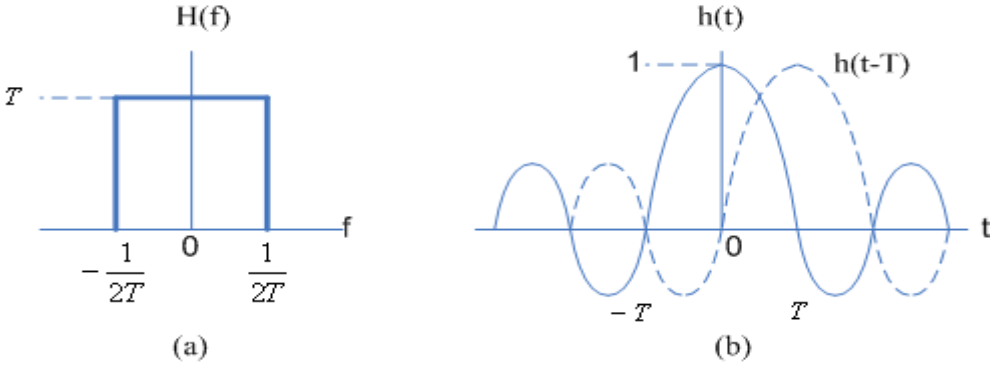

Figure 2.2: (a) Rectangular system transfer function. (b) Received pulse shape.

This definition provides an optimum solution in the sense of minimum required bandwidth for zero ISI to use the sinc pulse as the pulse shape in the time domain. That means the theoretical minimum bandwidth is $R_s / 2$ Hz to transmit R_s symbols per seconds. This happens while the system transfer function is rectangular, as shown in Fig. 2.2 (a). Its impulse response, the inverse Fourier transform of $H(f)$, is of form $h(t) = \sin c(t/T)$. Fig. 2.2 (b) shows how ISI is avoided at sampling instants with this signaling. Assuming two successive pulses, $h(t)$ and $h(t-T)$, both having long tails. The tails of $h(t)$ pass through zero at the instant $(t = T)$ when $h(t-T)$ is to be sampled, and all tails going through zero amplitude when any other pulse of the sequence $h(t-KT), k = \pm 1, \pm 2, \cdots$ is to be detected [3]. Theoretically, the rectangular function could be utilized as an ideal system transfer function to avoid ISI. In practice, however, a rectangular frequency-domain function results in large sidelobes in the time domain. Thus unless sampling is performed at exactly the correct sampling instants, the tail accumulated from others symbols will result in large ISI.

In order to reduce the effect from the sidelobes, H. Nyquist illustrated a solution that the transfer function does not have to be rectangular [4]. This specific transfer function $H(f)$ that results in zero ISI at sampling time instants is called the raised cosine filter. In Fig. 2.3, we can find that the bandwidth is no longer $W_0 = 1/2T$. The difference $W - W_0$ is called the "excess bandwidth", which means the extra bandwidth beyond the minimum required. The bandwidth of the raised cosine filter is calculated as:

$$W = (1+\alpha)R_s \tag{2.4}$$

where α is the roll-off factor and R_s is the symbol rate. The roll-off factor defines how much excess bandwidth is required.

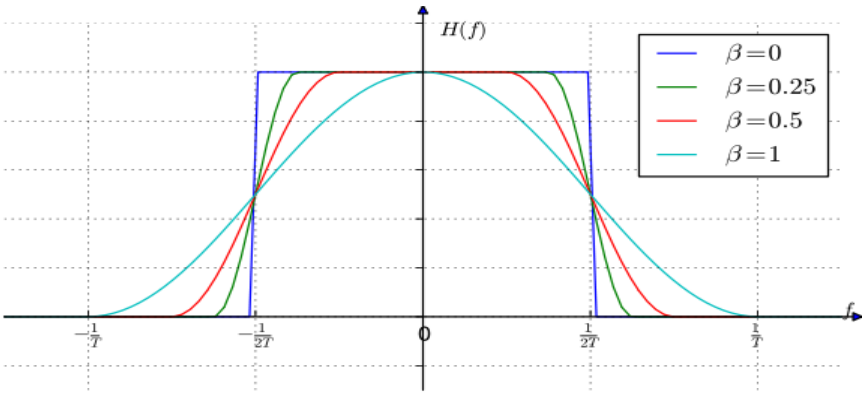

Figure 2.3: Frequency response of the raised cosine filter.

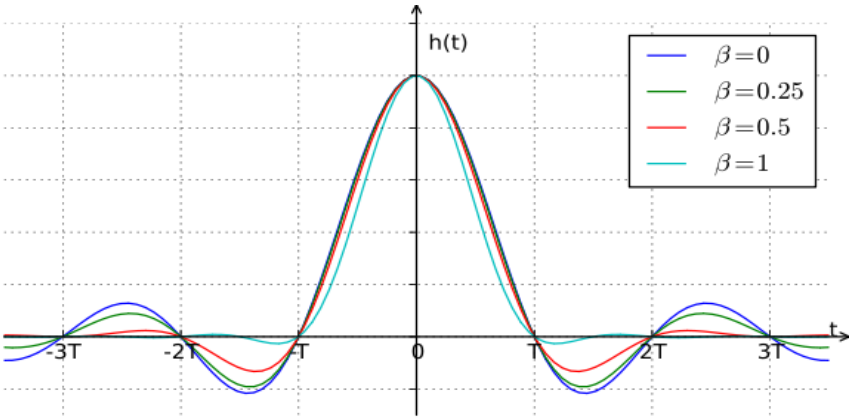

Figure 2.4: Impulse response of the raised cosine filter.

From Fig. 2.3 and Fig. 2.4, one can see that as α increases (therefore, bandwidth increases), the tail of the corresponding time domain waveform reduces. If the roll-off factor approaches 1, even if the sampling time is not so precise, ISI will not be serious due to the small tail. But it requires 100% excess bandwidth, which makes it bandwidth inefficient. If the roll-off factor equals 0, it follows the Nyquist minimum-bandwidth case, but the amplitude of the tail is much larger than the case with roll-off 1, which potentially makes the system vulnerable to timing errors. The smaller the roll-off factor, the less excess bandwidth. In a design, bandwidth efficiency and sensitivity to timing error should be jointly considered.

2.3 Square-Root Raised cosine filter

A matched filter in digital communication system is usually employed in the receiver. The overall frequency response of the system considering both the transmit filter and receive filter is:

$$H(f) = H_t(f)H_r(f) \tag{2.5}$$

where $H_t(f)$ and $H_r(f)$ represent the transmit filter and receive filter, respectively. In the case of a raised-cosine filter case, when $H_t(f)$ is matched with $H_r(f)$.

$$H_{rc}(f) = H_{rrc}(f) \cdot H_{rrc}(f) \tag{2.6}$$

$$\text{Or } \left|H_{rrc}(f)\right| = \sqrt{\left|H_{rc}(f)\right|} \tag{2.7}$$

where $H_{rc}(f)$ represents the raised cosine filter and H_{rcc} represents the square root raised cosine filter.

2.4 Modulation

Modulation is the process of varying one property of a high frequency periodic waveform such as the phase in phase-shift keying (PSK) signaling or the amplitude in pulse amplitude modulation (PAM). Modulation allows multiple levels, wand the level determines how many bits are carried by one symbol. However, as the level increases, PSK or PAM signal will be more sensitive to noise if the transmitted signal power is kept constant. Quadrature amplitude modulation (QAM) has the property of modulating both amplitude and phase, which meets the requirements of maximum noise immunity while power is kept close to minimum.

2.4.1 Phase Shift Keying

Phase-shift keying is the modulation that only the phase of the carrier wave varies. The way to vary the phase can be done through addition of the sine and cosine waves.

$$s_k(t) = \cos(\omega_0 t + \theta_k) = a_k cos\omega_0 t - b_k sin\omega_0 t \tag{2.8}$$

where $a_k = cos\theta_k;\ b_k = sin\theta_k.\ s_k(t)$ consists of two orthogonal signals, a sine and a cosine signal, with amplitude a_k and b_k. If different values are given to a_k and b_k, M-PSK can be regarded as the sum of two orthogonal M-ASK signals.

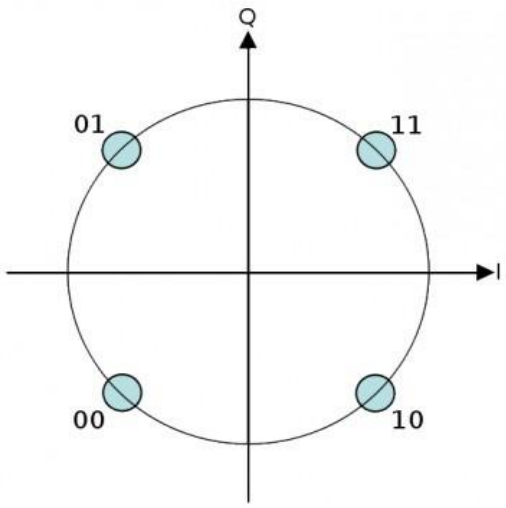

Figure 2.5: Constellation diagram of QPSK [4].

A general way to present the amplitude and phase properties of a signal in complex plane is use the constellation diagram. The real axis and imaginary axis represent the in-phase and quadrature components, respectively. The amplitude of the in-phase component shows the amplitude of the cosine wave, while the amplitude of the quadrature component is modulated by the sine wave.

In Fig. 2.5, the constellation of QPSK shows 4 points in $\frac{\pi}{2}$ phase difference. The symbol '11' represents that cosine and sine waves has the same amplitude of '1', which show the phase of symbol '11' as $\frac{\pi}{4}$, the symbol '01' represents the negative amplitudes of cosine and a positive amplitude of the sine wave, which results in a phase of $\frac{3\pi}{4}$.

2.4.2 Quadrature Amplitude Modulation

QAM, with both the carrier amplitude an phase vary, can be represented as:

$$s_k(t) = A_k \cos(\omega_0 t + \theta_k) \tag{2.9}$$

where A_k is the amplitude and θ_k is the phase. Eq. (2.9) can be rewritten as:

$$s_k(t) = A_k cos\theta_k cos\omega_0 t - A_k sin\theta_k sin\omega_0 t \tag{2.10}$$

where $X_k = A_k cos\theta_k$

$$Y_k = -A_k sin\theta_k$$

From these equations, $s_k(t)$ can be regarded as the sum of two orthogonal signals. In the special cases like 4-QAM, it performs the same as QPSK. In the case of 16-QAM, however, it is different from 16-PSK. 16-QAM is used here as an example to show why it is better than 16-PSK.

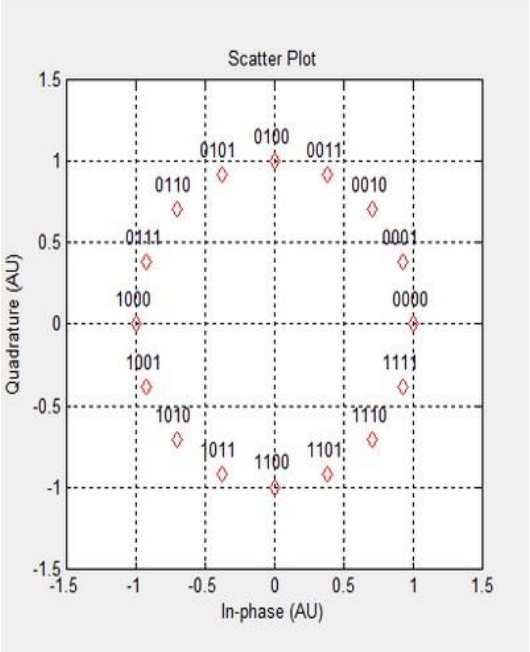

Figure 2.6: Constellation of 16-PSK [5].

In 16-PSK, assuming that the maximum amplitude of each point is A_m, which represents the distance between the origin and the constellation point, the distance of the two neighboring points equals:

$$d_1 = A_m(\pi/8) = 0.393A_m. \quad (2.11)$$

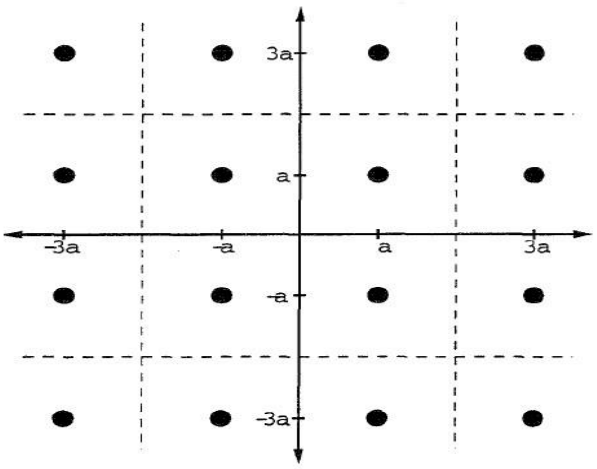

Figure 2.7: Constellation diagram of 16-QAM [6].

In the 16-QAM, the diagram shows that the amplitude of the in-phase and the quadrature components have 4 levels: 3a,-3a, a, and -a. The distance between the two neighboring points is:

$$d_2 = \sqrt{2}A_m/3 = 0.471A_m. \quad (2.12)$$

Comparing Eq. (2.11) and Eq. (2.12), we find that d_2 is 1.2 times bigger than d_1. Additionally, with a maximum amplitude for 16-PSK and 16-QAM, the average power of 16-QAM is much smaller.

2.5 Channel Frequency Response Analysis

The frequency response of the channel determines the output spectrum response for the input stimulus. The measurement of the magnitude and phase of output can be expressed as a function of frequency. It may be defined as the spectrum of the output signal divided by the spectrums of the input signals. For example, for a sine wave at a given frequency as the

input, a linear system will respond at that same frequency with a certain magnitude and a certain phase angle relative to the input. Thus, the difference between the two parameters, magnitude and phase, defines the frequency response.

The channel frequency response needs to be further discussed because the goal of this thesis is to reduce BER caused by attenuation and to exploit the broader frequency range of the channel. After further study and analysis, the special characteristic of the channel like the telephone line channel is potentially capable of delivering a high speed using the frequency higher than the range that is normal used. In this case, the attenuation of the channel in the frequency domain is flat until about 84.09 KHz and gradually increases after 84.09 KHz.

While the design is applicable for any severe ISI channels, the telephone line is the channel medium chosen for experiments. The telephone line supports delivering digital data to provide Internet access through technologies such as digital subscriber line (DSL) technology. The bit rate of DSL typically ranges from 256 Kbps up to 40 Mbps in the direction to customers, depending on the line condition or technology applied. In our experiment, the telephone line available in the lab is only 10 meters long. This does not represent a typical phone line channel.

Due to the limitation of the rate of the analog to digital converter (ADC) and digital to analog converter (DAC) available for experiments, the sampling rate cannot exceed 1 MHz. But the attenuation is negligible and unnoticeable within 1 MHz because of the short line length. To emulate the practical scenario better, a filter between the channel and the receiver is added and designed. Analysis indicates that a filter called maximally flat filter

was found to approximate the performance of a practical telephone line channel.

The maximally flat filter is a specific type of lowpass filter which has smooth and flat magnitude in the passband and stopband. Fig. 2.10 depicts the frequency response of the maximally flat filter. From Fig. 2.10, the range of the passband, which simulates the bandwidth of the telephone line, is between 0 Hz to 54.25 kHz (Normalized Frequency point 0.25). The attenuation is 3 dB for the cutoff frequency of 54.25 KHz. In the stopband, channel attenuation is smooth and gradually increases as frequency increases. At 84.09 kHz (Normalized frequency point 0.3875), the stopband loss is 20.24 dB.

Table 2.1: Magnitude loss induced by the maximally flat filter in the receiver.

Frequency [KHz]	Loss [dB]
54.25	3
84.09	20.24
151.9	165

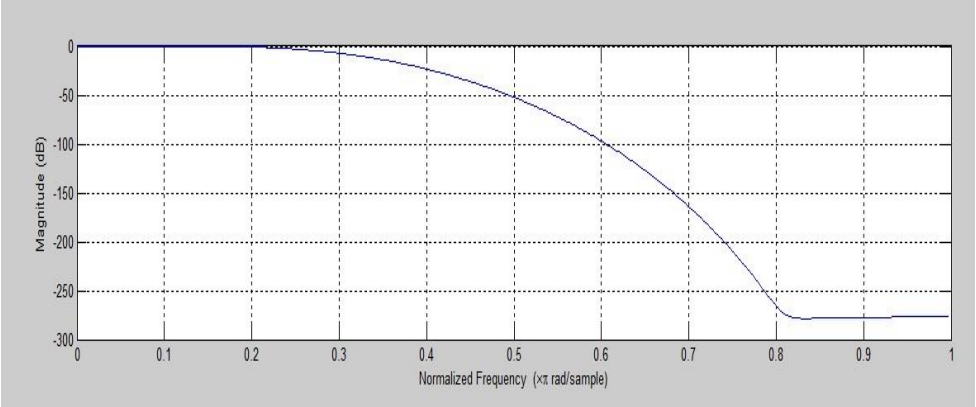

Figure 2.8: Frequency response of the maximally flat filter.

The phase response is the relationship between the phase of a sinusoidal input and the output signal after it passes through the system. The phase response of the maximally flat

filter is represented in Fig. 2.11, which shows that the response is linear.

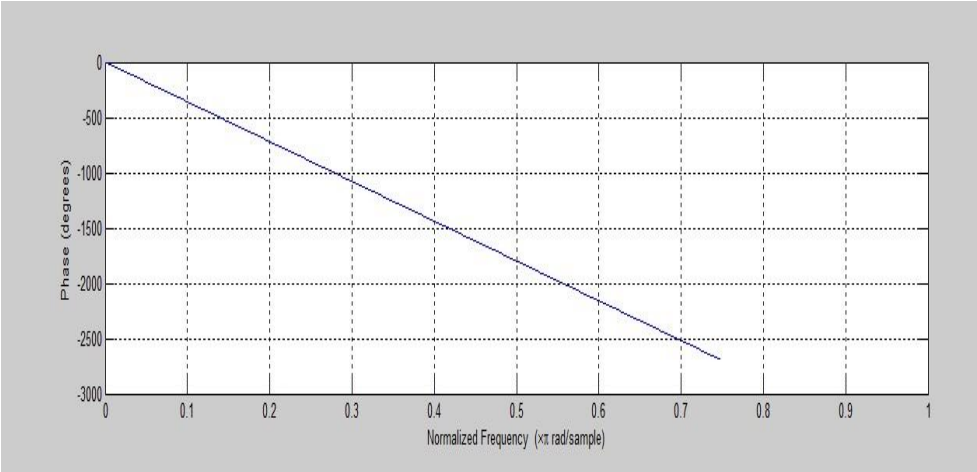

Figure 2.9: Phase response of the maximally flat filter.

The impulse response is the inverse Fourier of the channel frequency response. Fig. 2.12 shows the impulse response of the maximally flat filter. The maximal value of impulse response in this case is 0.29 at time 46us.

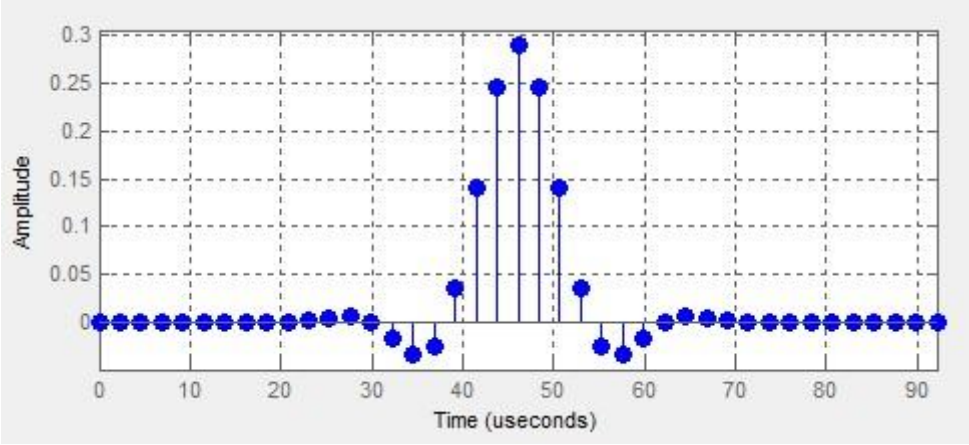

Figure 2.10: Impulse response of the maximally flat filter.

2.6 FPGA

The current mainstream of integrated circuit (IC) verification is based on using hardware description language like Verilog or VHDL to create a high level of representation of the register-transfer level (RTL) circuit after mapping and routing. A field programmable gated array (FPGA) is a device that can be programmed.

The basic logical component of an FPGA is called ‘logic block’, which can be configured to be interconnected, which allows the blocks to be wired together. This is somewhat like designing a virtual testbench into one chip and revising it at any time. The logical blocks can perform complex combinational logical functions or simple logic gates like ‘OR’ and ‘NAND’. In the most current FPGAs, the programmable logic block also contains memory function like flip-flop or other memory block [5]. FPGA can be used to implement complex DSP algorithms such as filtering of the signal, to implement error correction codes. Application specific integrated circuit (ASIC) is an integrated circuit for particular uses. The programmable feature makes FPGA suitable for a variety of projects in the early phase, whereas ASIC is typically used after a design is final. In terms of manufacturing cost, ASIC design is quite high because the cost of PCB layout and fabrication materials, unless made in large quantity. In the early stage of system developments, it’s risky to invest a huge amount of money in the ASIC layout. An FPGA board integrated with other chips can achieve most of the functions of ASIC. The most important feature of FPGA based systems is that it is flexible to fix defects through reprogramming rather than going through the lengthy IC process. This makes FPGA a preferable alternative than ASIC for a large number of applications [5].

2.6.1 Virtex5 XC5VLX110T

The Virtex series of FPGAs have integrated features that include FIFO and ECC logic, DSP blocks, PCI-Express controllers, Ethernet MAC blocks, and high-speed transceivers. In addition to FPGA logic, the Virtex series includes embedded fixed function hardware for commonly used functions such as multipliers, memories, serial transceivers and microprocessor cores [7]. The Virtex5 family provides many IP-system level blocks, including 36-Kbit block RAM/FIFOs, second generation 25×18 DSP slices, Select IO technology with built-in digitally controlled impedance, system monitor functionally, enhanced clock management tiles with integrated DCM and Phase Lock Loop (PLL) clock generator and advanced configuration option. PowerPC® 440 microprocessor embedded blocks allow advanced logic designers to build the highest levels of performance and functionality into their FPGA-based systems. Up to 330,000 logical cells includes 207,360 internal fabric flip-flops with clock enable, 207,360 real 6-input look-up tables (LUTs), Logic expanding multiplexers and I/O registers. The Virtex 5 family supports up to 550MHz clock, up to 6 clock management tiles each containing 2 DCM and 1 PLL [7].

Figure 2.11: Virtex-5 XC5VLX110T.

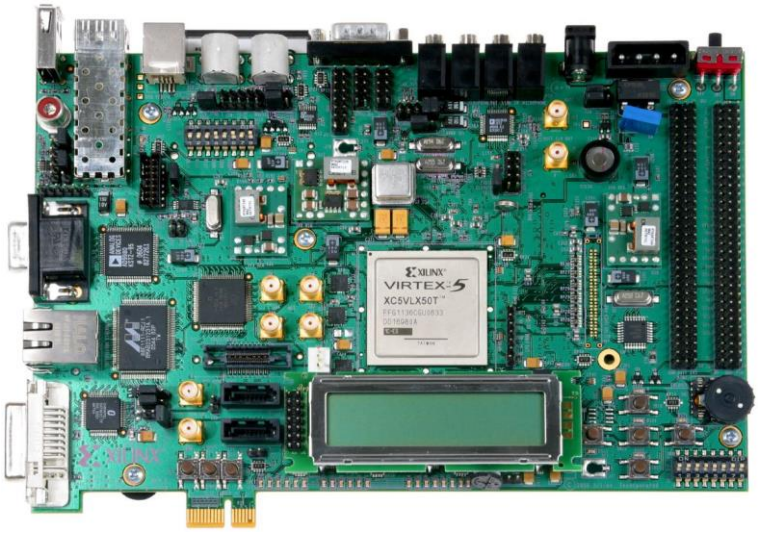

Figure 2.12: Xilinx ML505 development board.

ML505 is a general purpose FPGA development board integrated with Virtex5, which provides serial connectivity, signal processing and embedded processing resources. The abundant interface helps FPGA to connect with other equipment and expand the function that FPGA can handle. USB keyboard allows computer to connect with board. Ethernet port and an RJ-45 Ethernet cable help the board to link with the Internet to do simulation about Network. Computer speaker with audio cable realize the function of audio signal processing even FPGA can work as MP3 player or better. Platform emulation board can store the data or programming bitstream in Compact Flash (CF) card to avoid reprogramming each time when restart.

2.6.2 Programming with Verilog

Different FPGA manufactures have their own software development tools for FPGA design. The ISE is developed by Xilinx for synthesis and behavior simulation of HDL language, timing analysis, map and route, and configuration of the target device to program. The ISE tool package contains ChipScope, which is used as a logical analyzer to monitor internal signals in the chip, Xilinx Platform Studio (XPS) helps configure hardware and connect the

embedded processor-based system.

Verilog, or VHDL, is the standard hardware description language used to model electronic components. It is commonly used in digital circuits at register transfer level of abstraction. The Verilog is similar to C language, but it is a dataflow language, which means it executes multiple instructions simultaneously. Compared with VHDL, Verilog is easier to learn.

The structure of hardware can be divided into several modules. In Verilog, one has to define each module by its input and output. Each module is in RTL, thus it is controlled by clock and hardware registers to store data. In Verilog, one also needs to define bit and port data type. Reg is used to define the variable for writing while Wire is to define the signal for reading. Here we see one small example programmed by Verilog.

```
module addsub (a, b, c, clk, result);
                input[7:0] a;
                input[7:0] b;
                input c;
                input clk;
                output[8:0] result;
                reg[8:0] result;
               always @(posedge clk)
               begin if (c)
                    result = a + b;
              else result = a - b;
              end
```

endmodule

In this case, input ‘clk’, ‘a’, ‘b’, ‘c’ and output ‘result’ are defined first. The ‘always’ clause illustrates the trigger condition, as the positive edge of the clock in the example. The ‘If’ clause works the same as in C language. If ‘c’ equals 1, then runs the next statement; otherwise it jump into ‘else’ statement.

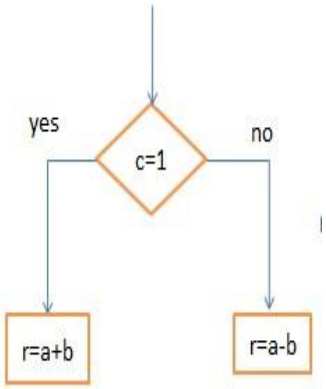

Figure 2.13: Data flow of module ‘addsub’.

Fig. 2.15 shows the data flow of Module ‘addsub’. If ‘c’ equals one, then add ‘a’ and ‘b’ then return into ‘result’; If zero, the result will equal the subtraction of the two inputs.

Before verification of the previous Verilog code, writing a testbench is an essential process. Modelsim is the software product from Mentor Graphic which provides the debugging capabilities of Verilog codes. The structure of the testbench is the same as a normal module, but has no input or output. It provides the ideal test environment when you set clock cycle period and input which verify functionality.

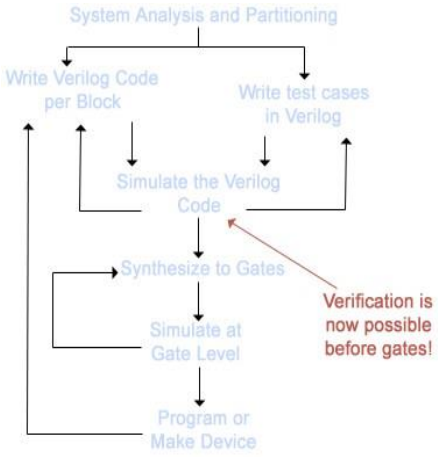

Figure 2.14 Data flow using Verilog.

Fig. 2.16 presents the basic flow diagram for the verification of Verilog. After programming Verilog code for block and testbench, the simulation can be done through Modelsim and examined the code by analyzing the waveform. The testbench simulates test environment and waveform would exhibit the output under any given input.

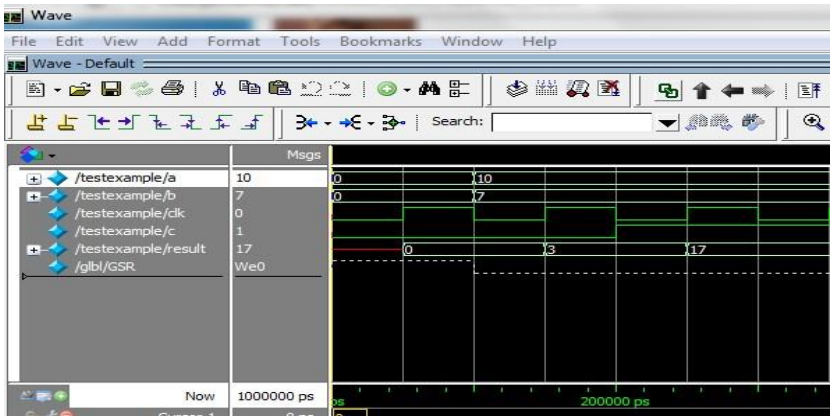

Figure 2.15: Modelsim simulation result of testbench.

Fig. 2.17 shows the waveform simulated from Modelsim; it collects the result from testbench. The input ‘a’ and ‘b’ have been assigned values of 10 and 7, respectively. If input ‘c’ is 0, then the function of subtraction is verified from ‘result’ of 3. While ‘c’ turns to 1,

17 proves that the add operation works correctly. After verifying through waveform from testbench, it is concluded that Verilog code for Module ‘addsub’ achieves the target function and is available to move forward to the next step ‘synthesis’.

2.6.3 Synthesis

Synthesis is the process of transferring the abstract circuit behavior, like RTL described by Verilog, into the design implementation of logical gate. After synthesis, ISE will return a complete synthesis report which includes the device utilization summary, partition resource summary and timing analysis. In the device utilization summary, it lists the number of LUT, I/O and DSP utilized in this design. If the design requires more than what the FPGA could provide, then ISE reports an error during synthesis. In the timing report, ISE automatically finds the shortest combinational delay that restricted the minimum period. The minimum period is the least time required to ensure the signal from input to output, which defines the maximal clock time.

2.6.4 Implement design

The previous section about Verilog code and synthesis seems to be irrelevant of FPGA, but the next step, ‘Implement Design’ is more crucial because the design has to follow the required clock speed in FPGA and find the appropriate pin as output. Implement design includes ‘translate’, ‘map’, ‘place and route’, and ‘generate programming file’. ‘Translate’ is the process of merging the incoming netlists and constraints into a Xilinx design file. ‘Map’ fits the design into the available resources on the target device, and optionally, places the design. ‘Place and Route’, literally place and route the design to the timing constraints. ‘Generate programming file’ creates a bitstream file that can be

downloaded and is recognized by the device.

There are two ways to finish ‘implement design’ work, which are set through Constraints editor and PlanAhead or made User Constraint File (UCF), respectively. The Constraints Editor embedded in ISE has provided a convenient way to design timing constraints. In the window of Timing Constraint, Clock domain automatically finds the unconstrained clock and sets the period to define the each clock cycle. At the Inputs and Outputs window, offset in and out times are to be set in external setup time and external clock pad. The Offset In Before (Pad to Setup) constraint allows the external clock and external input data to meet the setup time on the internal flip-flop. The Offset Out After (Clock to Out) constraint gives more control over the setup/hold requirement of the downstream devices and with respect to the external output data pad and the external clock pad [8]. Offset In and Offset Out constraints with clock period are shown in Fig 2.18 and Fig. 2.19, respectively. PlanAhead is another embedded design tool which helps users to assign the input and output signal into specific FPGA pins. It also allows the user to set the voltage standard of pins so as to meet the variable environment. The block diagram of each board can be downloaded from the Xilinx website and it reveals the internal connection of each pin so that we can utilize the various I/O of board or other integrated functions like video processing through IP core. After timing constraints and pin assignment, all implement design work has finished and ISE will automatically generate the UCF file which includes constraint information required above. Manual solution is another direct method to understand the programming syntax of UCF and write it in directly rather than assistance through other software.

After all the previous is set, with one click of the implement design icon, ISE will run

'translate', 'map' and 'place and route' in sequence. It might take a long time to complete depending on complexity of the design. Finally, the bitstream file will be generated automatically and is delivered to and recognized by the board.

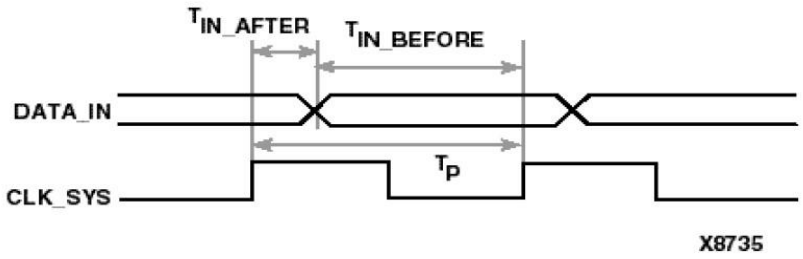

Figure 2.16: Timing reference diagram of offset in constraint.

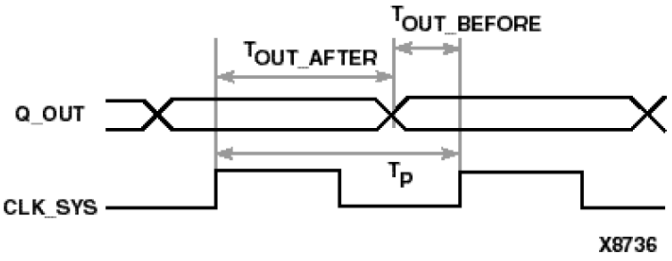

Figure 2.17 Timing reference diagram of offset out constraint.

2.6.5 Configure Target device

With one click of the icon of Configure Target Device, a popup window like Fig. 2.20 jumps out and after a. right click at the blanket in the center of the window, the device is added. The popup window 'Assign New configuration File' as shown in Fig 2.20 shows how to program bitsteam file into FPGA. Fig. 2.21 shows that 5 chips with Xilinx sign are connected as a chain. The first 4 chips are other CPLD or ROM, while the one with 'xc5vtx110t' is the target FPGA chip. By right clicking it and clicking 'program', the bit file

is programmed into the FPGA. Then we can use Chipscope to test the design.

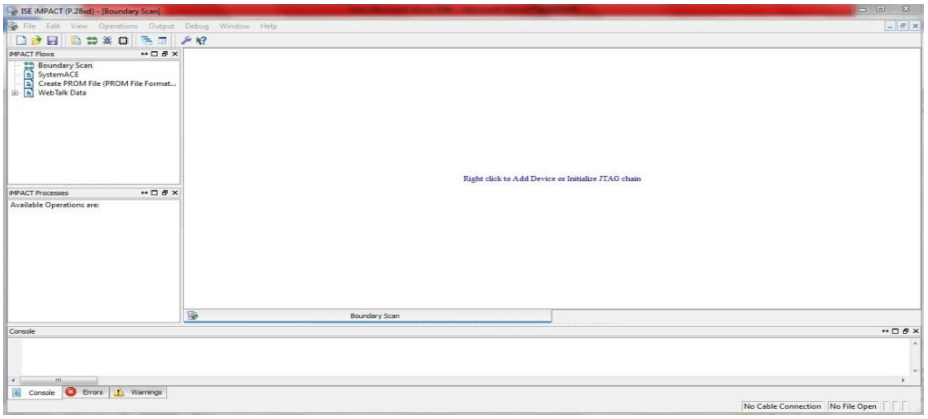

Figure 2.18: ISE IMPACT [boundary scan]

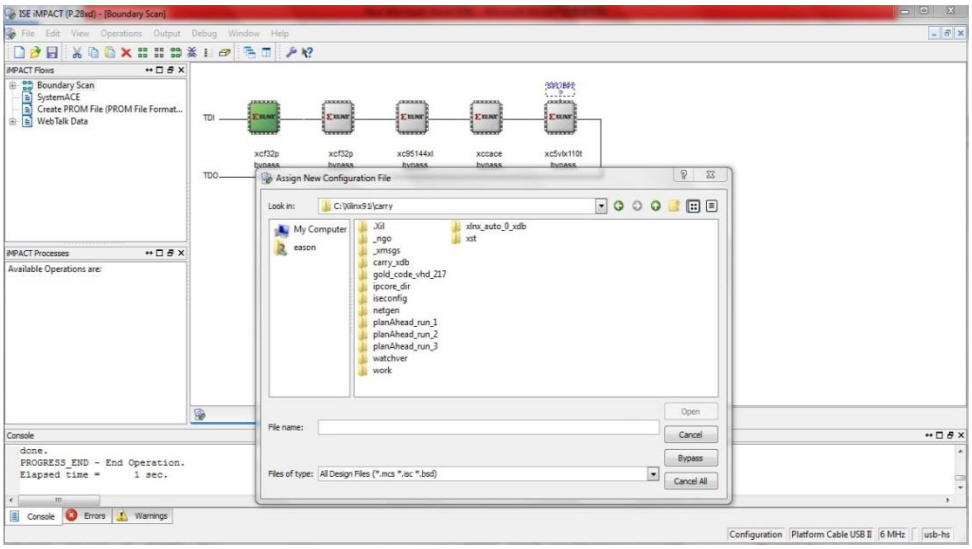

Figure 2.19: Assign new configuration file

2.6.6 ChipScope Vertification

ChipScope inserts the logic analyzer software core, which allows users to explore the internal signal of FPGA. The ISE package includes the ChipScope Pro for convenient debugging. Signals are captured in the system at the speed of operation and brought out through the programming interface (JTAG). Two IPcores had been inserted into the design as module before synthesis. One is called chipscope_icon and another is chipscope_ila. The

chipscope_icon (Integrated Controller Core) provides an interface between the JTAG Boundary Scan (BSCAN) interface of the FPGA device and the ChipScope Pro cores, including the following types of cores: Integrated Logic Analyzer (ILA), Virtual Input/Output (VIO) [9]. The ILA core (Integrated Logic Analyzer) is a customizable logic core that can be used to monitor any internal signal of a design. The ILA core has similar features as the logic analyzer that enables user to set the trigger sequence and storage qualification. It has user-selectable trigger width, data width, and data depth. It also has multiple trigger ports, which can be combined into a single trigger condition or sequence. The storage qualification option allows the core to store a sample only when a certain condition is met. Overall, the ChipScope is a strong virtual logic analyzer that saves large amounts of time for testing and debugging.

After setting IPcore, double click ‘Analyzer Design Using ChipScope’ and Chipscope will open automatically. Run ‘Open Cable/Search JTAG Chain’ and ChipScope will search the trigger condition and store the signal of the trigger port. Fig. 2.22 presents the waveform of the detected signal. The difference between ChipScope and Modelsim is that Modelsim does the simulations but ChipScope captures real signal detection.

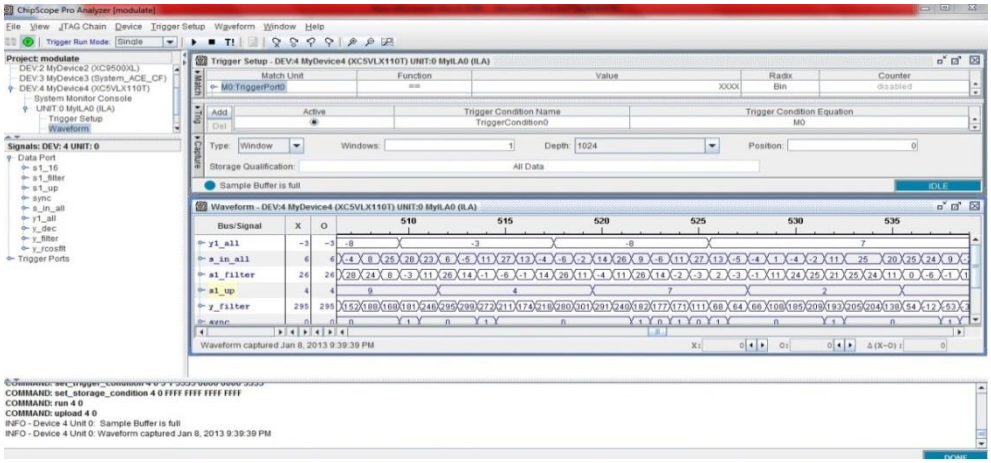

Figure 2.20: ChipScope pro analyzer.

3. High Speed Digital Transmission over Severe ISI Channels

To access how severe ISI due to channel response is at three different data rates, the received signals are simulated in the absence of additive white Gaussian noise (AWGN). The eye diagrams of the signals with three different rates passing through the channel are illustrated in Fig. 3.1. The top left picture of 60 KHz case shows no ISI effect. The 80 KHz case experiences much worse ISI condition than the 60 KHz case due to the 30dB attenuation due to the channel. It becomes obvious that the signal achieving the target BER without an equalizer at 150 KHz is impossible from the closure of eye pattern shown in Fig. 3.1(c).

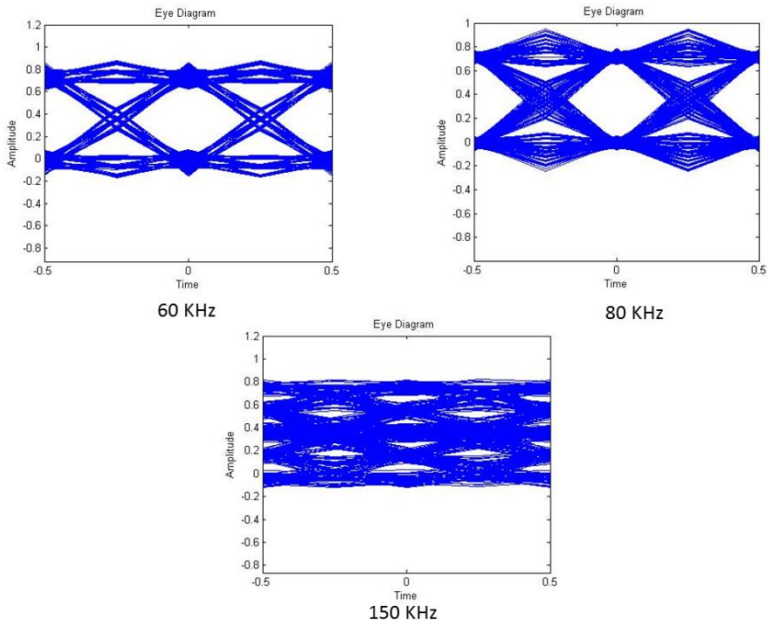

Figure 3.1: Eye diagram of signals in different frequency.

3.1 Simulation Component

3.1.1 Upsampling factor

According to the Nyquist sampling criterion, the sampling rate should be at least two times the bandwidth to avoid aliasing. Upsampling is the procedure to increase the sampling rate in discrete time domain. The upsampling factor should be an integer or a rational fraction greater than the sampling unit. This factor multiplies the original signal rate to achieve the sampling rate. It is important to apply an upsampling factor in constructing the raised cosine filter.

Consider a discrete signal $f(k)$ and "L" denotes the upsampling factor:

Definition of upsampling: $$g(k) = \begin{cases} f\left(\frac{k}{L}\right); & if \frac{k}{L}\ is\ an\ integer \\ 0; & otherwise \end{cases} \quad (3.1)$$

Table 3.1 below lists the corresponding symbol rates and data rates in different upsampling factors.

Table 3.1: Data and symbol rate versus upsampling factors

Upsampling Factor	Symbol Rate	Data Rate		
		BPSK (1bit/symbol)	QPSK (2bits/symbol)	16-QAM (4bits/symbol)
8	4 Ksps	4Kbps	8 Kbps	16 Kbps
16	2 Ksps	2 Kbps	4 Kbps	8 Kbps
32	1 Ksps	1 Kbps	2 Kbps	4 Kbps

The process of upsampling can be divided into 2 steps. First, zero samples will be inserted

between two neighboring signals, also called zero padding. The number of zero samples is *N-1*, where N is the upsampling factor. For example, the discrete signal sequence $x[n]$ given as $\{4,5,4,7,6,3\}$ shows in Fig. 3.3 (a). After zero padding with the upsampling factor 3, the sequence turns into v[n] $\{4,0,0,5,0,0,4,0,0,4,0,0,7,0,0,6,0,0,3,0,0\}$, depicted in Fig 3.3[b]. Then *v[n]* is low-pass filtered resulting in Fig. 3.3(c).

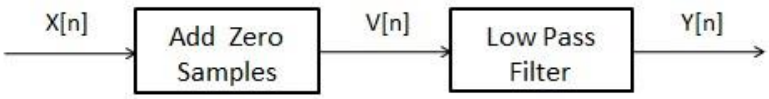

Figure 3.2: Block diagram of upsampling steps.

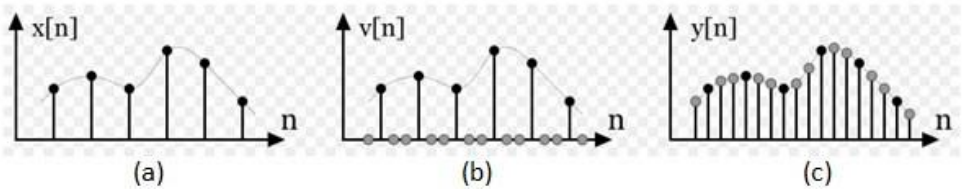

Figure 3.3: Illustration of upsampling steps in the time domain.

Fig. 3.4 shows each step of upsampling in the frequency domain. Fig. 3.4 (a) shows discrete-time Fourier transform of sequence $x[n] = x_c(nT)$. Now the upsampling factor is 3, resulting in the sequence $v[n] = x[n/3] = x_c(nT/3)$, whose discrete-time Fourier transform is plotted in Fig. 3.4 (b). Fig. 3.4 (c) shows the upsampled signal in frequency domain after traveling the lowpass filter with the cutoff frequency $\frac{\omega_m}{3}$.

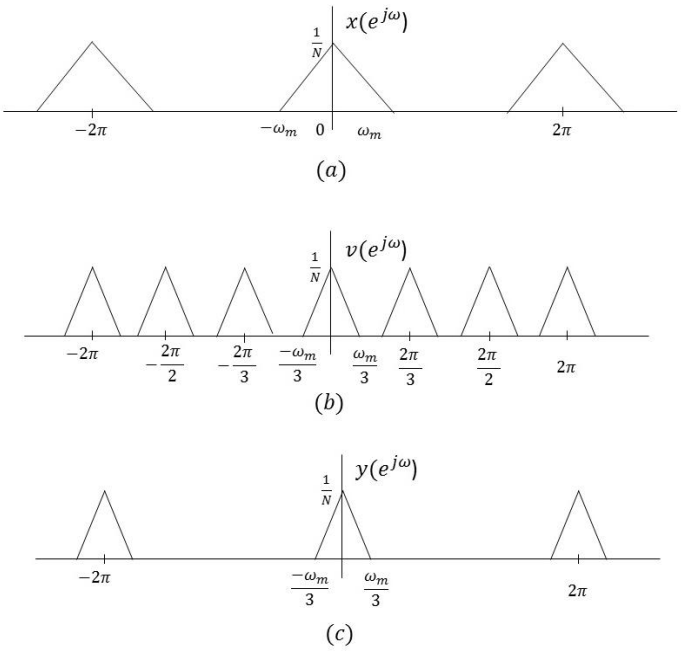

Figure 3.4: Frequency domain illustration of upsampling.

3.1.2 Filter Design

As discussed in Section 2.3, a square root raised cosine filter is applied in both the transmitter and the receiver to satisfy the Nyquist criterion for zero ISI and to maximize the received signal-to-noise ratio. Two parameters need to be analyzed for filter design: sampling rate and roll-off factor. The upsampling factor varies with the symbol rate for both the baseband and the other subbands so as to maintain the same sampling rate. Another core parameter is the roll-off factor. As discussed before, a factor results in robustness to sampling timing errors, extra bandwidth and power consumption.

Another filter to be designed is the maximally flat filter. From Section 2.5, the combined effect of the short telephone line and the maximally flat filter will lead to the amount of ISI, that a normal phone line of 1km length might produce when used for high-speed signaling.

3.1.3 D/A converter and A/D converter

DAC and ADC are a pair of devices that convert the signal between continuous physical quantities and digital numbers that typically represent the quantity's amplitude. One converter can be regarded as performing the reverse process of the other.

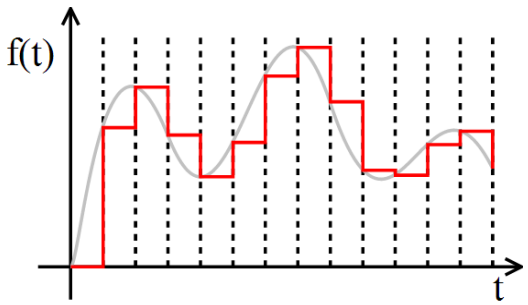

Figure 3.5: Piecewise constant output of a conventional DAC [11].

A DAC converts finite time series data into a continually varying voltage. The input digital number updates the analog voltage at uniform sampling intervals. Usually the output value changes from the previous value to the one represented by the current latch number, typically within a clock cycle, which causes each number to be latched in sequence. Fig. 3.5 shows the operation of a conventional DAC.

The resolution of an n-bit ADC is a function of how many pieces the signal can be divided into. The formula to calculate the resolution is 2^n. For instance, an 8 bit ADC can encode an analog input to one in 256 different levels, since $2^8 = 256$ [10]. Therefore, the best resolution is 1 part of 256, or 0.3906% of full scale.

The ADC and DAC used in experiment are purchased from Digilent. The package of PmodAD1 includes two AD7476A 12-bit A/D converter chips, one 6-pin header connector, one 6-pin connector and two 2-pole Sallen-Key anti-alias filters. The two ADC chips and

filters are packaged in one circuit. The AD7476A converts an analog signal ranging from 0-3.3volts to a 12-bit digital value in the range of 0-4095. The AD1 has two simultaneous A/D conversion channels, each with a 12-bit converter and a filter. The AD1 can also convert a single stream of analog signals using only one channel [12]. The analog signal range supports FPGA I/O pin volt standard. The serial bus can run at a rate of 16MHz and each 16 clock cycles generates one output. Thus, the throughput rates of ADC are up to 1MSPS. That means the maximal sampling rate is 1MHz.

Fig. 3.6 presents the timing diagram of ADC. While $\overline{CS}$ (chip select) is active at low logic input, the following 16 falling edge of SCLK is essential to complete one conversion and access the conversion result as SDATA. Once a data transfer is complete (SDATA has returned to three-state), another conversion can be initiated after the quiet time, t_{quiet}, has elapsed by bringing $\overline{CS}$ low again [13] .

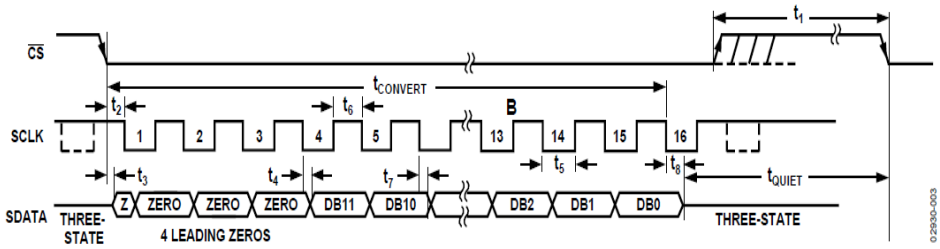

Figure 3.6: AD7476A serial interface timing diagram [14].

The package of Pmod-DA1includes two AD7303, which are dual, 8-bit voltage out DACs that operate from a single +2.7 V to +5.5 V supply [15]. The 8-bit means an 8 bit input in series to complete one conversion. This chip supports clock rates up to 30MHz.

Fig. 3.7 depicts the timing diagram of AD7303. Synchronous signal $\overline{sync}$ works same as

$\overline{CS}$ in AD7476A case. The low $\overline{sync}$ enables 16-bit to be continuous written into register and to complete one conversion. The input shift register content DIN (data input) contains 8-bit control and data bits. The control bit provides feature-like internal/external reference, power up/down mode and output pins selection. DB (data bit) 15 selects between internal or external reference signal. DB14 does not assign any commitment. DB13 loads DAC bit for synchronous update of DAC outputs. DB12 selects power-down DAC B and DB11 actives Power-down mode for DAC A. DB10 is used to select either DAC A or DAC B. DB9 is used in conjunction with CR0 to implement the various data loading functions with DB8 in conjunction with CR1. DB7–0 contains the data used to update the output of the DACs. DB7 is the MSB (Most Significant Bit) and DB0 is the LSB (Least Significant Bit) of the 8-bit data word.

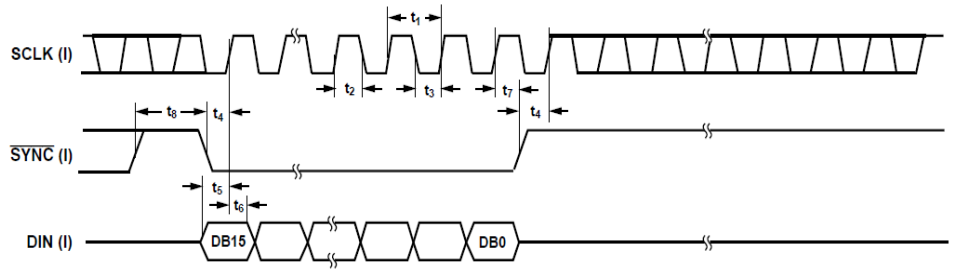

Figure 3.7: AD7303 timing diagram [15].

3.2 Design of the Single-band Architecture

The single-band architecture is illustrated in Fig. 3.8 with all the blocks of the design included. First, the signal should be upsampled and then filtered so as to reconstruct the desired frequency range. Then, the DAC helps to translate the digital signal into an analog waveform to pass through the channel. After received and converted from the ADC, the

manually made maximally flat filter will introduce additional attenuation to simulate the desired channel response for the received signal. Then another square root raised cosine filter does the rest of the matched filter's work at the receiver. Finally, the baseband signal is downsampled and demodulated.

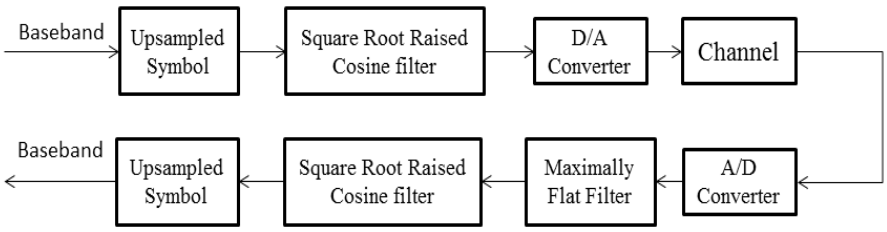

Figure 3.8: Block diagram of single-band.

The data rate for the single-band architecture is 217 Kbps with BPSK, and the matched filter designed for transceiver is a square root raised cosine filter with a roll-off factor 0.4. The upsampling factor is designed to be 2 to meet the minimal requirement according to Shannon Sampling Theorem. The total bandwidth is defined as half of the symbol rate and the excess bandwidth due to the raised cosine filter. Thus, the bandwidth of this case is calculated as $217 \times \frac{1+0.4}{2} = 151.9$ KHz. The parameters of the design are listed in detail in Table 3.2.

Table 3.2: Parameters for the single-band architecture

	Baseband
Modulation	BPSK
Symbol rate	217k
Sample rate	2

Data rate	217K
Sampling rate	434K
Roll-off factor	0.4
Total bandwidth	151.9K

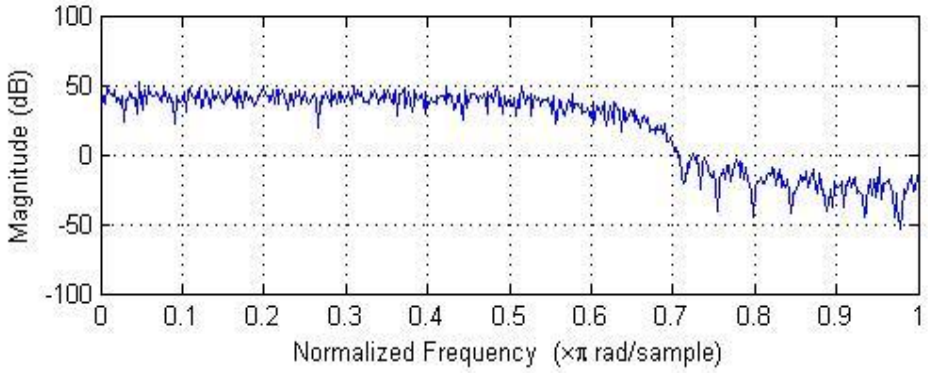

Figure 3.9: Frequency response of the raised cosine filter in thesingle-band architecture.

The cutoff frequency is 108.5 KHz, as shown a Normalized Frequency of 0.5 in Fig. 3.9. After the cutoff frequency, the attenuation increases until approximately the normalized frequency of 0.7. The symbol rate is 108.5 KHz and requires an excess bandwidth of 43.4 KHz. So, the total bandwidth is the sum of half the symbol rate and the extra bandwidth, equaling 151.8 KHz.

3.3 BER Simulation Results of the Single-band Architecture

The goal of the design in this thesis is to compare design options that both transmit the data rate of 217 Kbps. With BPSK modulation, the symbol rate is equal to the data rate of 108.5 KHz. This architecture occupies a frequency range from 0 to 151.9 KHz. The BER with

this architecture is found to be 16.74%.

Although BPSK is more resistant to noise than the higher-order modulations (such as M-QAM at the same received power level), the BER is still unacceptable since the attenuation at the highest frequency is 130dB, causing severe ISI.

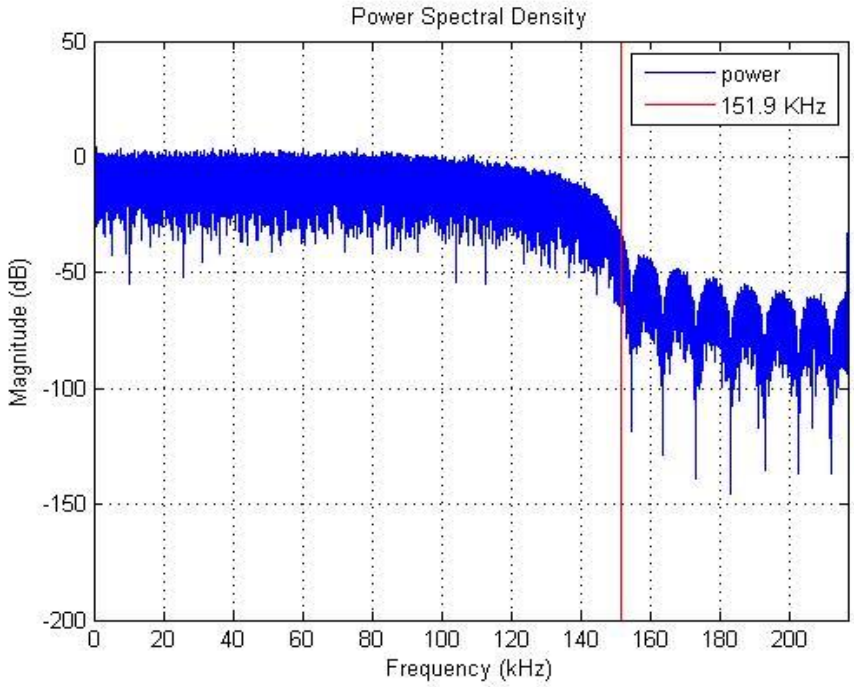

Figure 3.10: The power spectrum of the transmitted signal.

The power spectrum of the transmitted signal and the received signal after the channel are shown in Fig. 3.10 and Fig. 3.11 respectively.

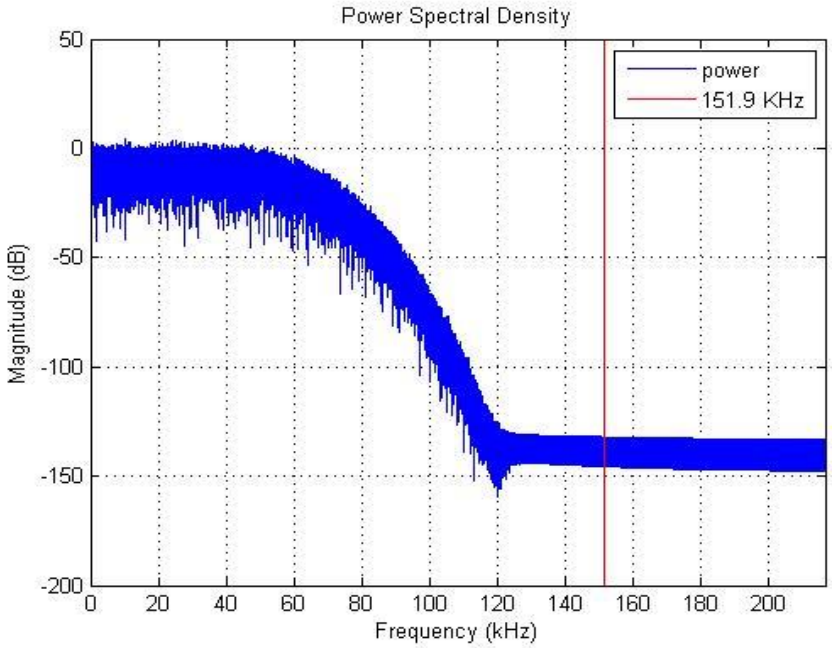

Figure 3.11: The power spectrum of the received signal.

4. Optimization of High Speed Digital Transmission over Severe ISI Channels

There are several parameters that need to be further discussed before the four-band architecture can be constructed. The first is the selection of a proper frequency range. If the frequency response of this range is smooth enough, ISI could be mild to result in an acceptable BER at a high SNR. The second parameter is the bandwidth of each subband and the guard band. The narrower each subband, the smaller the total bandwidth to avoid high attenuation at very high frequency. The guard band between each two neighboring bands should be narrow but sufficient to avoid inter-band interference. The third area is an appropriate modulation for each subchannel. To increase transmission speed than binary modulation, higher-order modulation schemes could be adapted to transmit more bits per symbol. A higher order modulation results in more bits per symbol, but higher order modulation is also more vulnerable to noise than lower order modulation.

This chapter presents a four-band design with the application of higher-order modulation and the raised cosine filter, which makes data transmission more efficient and achieves a lower BER compared than the single-band approach. Although a four-band scheme is optimized and tested, the general approach to optimize the multiband architecture could be used for any number of subband cases.

4.1 System Blocks

4.1.1 Filter design

The filter used to simulate the ideal channel response is split into real and artificial parts. The artificial part is the same as the one used in the single-band approach with a cutoff frequency of 54.5 KHz.

4.1.2 Carrier Demodulator

The mixer plays an important role in modulation process because it is used to shift the signal typical from baseband to the carrier frequency.

In general, baseband transmission does not require a mixer, but all the other subbands in the higher frequency range require a mixer and an up-converter to shift the signal from baseband to passband. The basic principal of mixing and up conversion is best explained via Fourier transformation. A time-domain signal passing through a mixer can be expressed as:

$$x_m(t) = x\,(t) \cdot \cos(2\pi f_T t), \text{ or } \quad x_m(t) = x(t) \cdot e^{j2\pi f_T t} \tag{4.1}$$

where "$x_m(t)$" represents the modulated signal and "$x(t)$" is the baseband signal, and "f_T" is the carrier frequency. The implementation of the down converter is the reverse process of the mixer.

4.1.3 QAM Demodulation

The demodulator of QAM consists a down converter, a lowpass filter (LPF), decision device and a multiplexer. Fig. 4.1 shows the demoudaltion process. The input signal S(t) can be expressed as:

$$\mathrm{S}(t) = \mathrm{m}(t)\cos(2\pi f_T t + \theta) \tag{4.2}$$

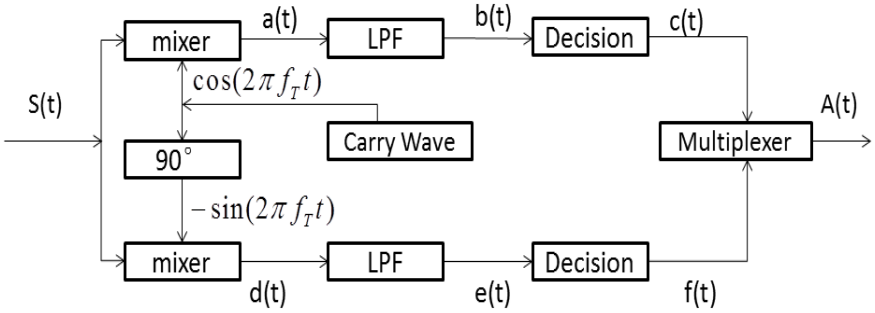

Figure 4.1: QAM demodulation scheme diagram.

The parameter "θ" represents the initial phase, and $m(t)$ is the received carrier-modulated signal.

The modulated signal S(t) multipies the carrier wave and 90 degree phase shifted of it.

The signals a(t) and d(t) as shown in Fig. 4.1 are expressed as:

$$a(t) = m(t)\cos(2\pi f_T t + \theta)\cos(2\pi f_T t) \tag{4.3}$$

$$= \frac{1}{2}m(t)[cos\theta + \cos(4\pi f_T t + \theta)]$$

$$d(t) = m(t)\cos(2\pi f_T t + \theta)\sin(2\pi f_T t) \tag{4.4}$$

$$= \frac{1}{2}m(t)[sin\theta + sin(4\pi f_T t + \theta)]$$

After passing through the LPF, the high frequency components in Eq. (4.4) and Eq. (4.5) are filtered out, resulting in baseband signals b(t) and e(t) expressed as:

$$b(t) = \tfrac{1}{2}m(t)cos\theta \tag{4.5}$$

$$e(t) = \tfrac{1}{2}m(t)sin\theta \tag{4.6}$$

The decision device divides the constellation into decision regions, and then uses these regions to identify the value of the received signal. The decision device needs to decide the magnitude for both the in-phase and quadrature components. The multiplexer, which is also

known as the data selector, chooses the in-phase or quadrature component and forwards the selected signal into one line with an equal chance to recover the transmitted bits. Fig. 4.2 shows the schematic of the 2-to-1 multiplexer and how it works as the data selector.

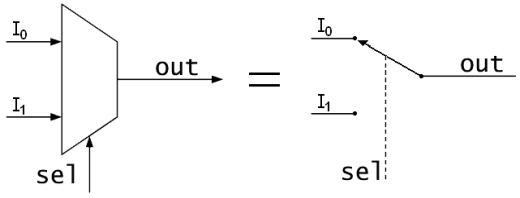

Figure 4.2: Schematic of 2-to-1 multiplexer.

4.2 Design of the Four-band Architecture

The implementation of four sub-channels with narrow guard bands should minimize the total bandwidth to avoid the high attenuation by the channel in the very high frequency range. The modulation method is QPSK for the baseband and 16-QAM for the other three subbands. The term “baseband” refers to the subband covering the lowest portion of the frequency range, rather than refers to the common baseband signaling.

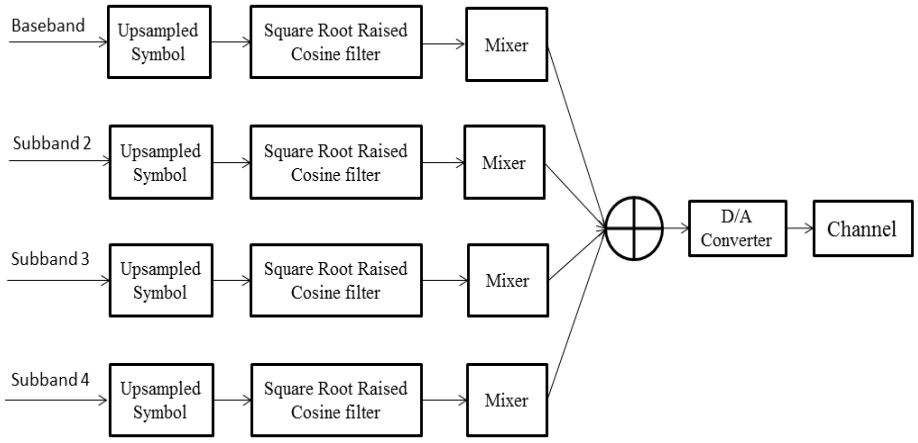

Figure 4.3: Block diagram of the four-band transmitter.

The block diagram of the transmitter is shown in Fig. 4.2, and the receiver is depicted in Fig. 4.3. Each band is paired with its specific square root raised cosine filter, which should be implemented to match the upsampling factor of the sub-channel. The D/A converter and the A/D converter are connected at the both ends of the channel.

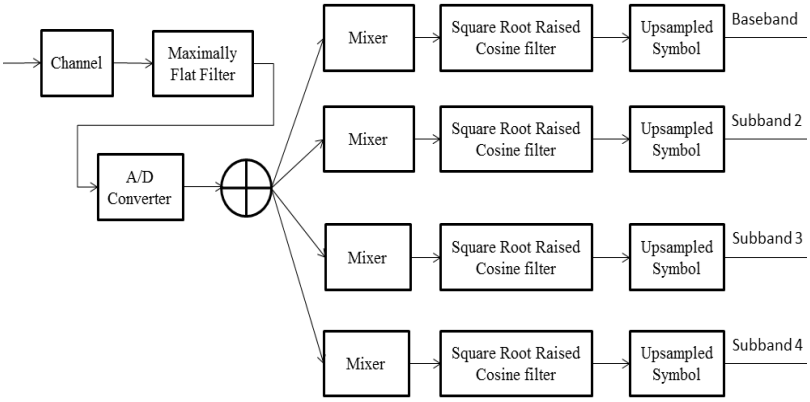

Figure 4.4: Block diagram of the four-band receiver.

After numerous subband and filter design attempts, the desired four-band architecture has been more or less optimized. The total bandwidth of the system is limited to 84.088 KHz. At this frequency, the channel has an attenuation of 20 dB. The parameters of this architecture are listed in Table 4.1. From this table, the total data rate for this design sums up to 16 Kbps. The approach to define the proper filter for the baseband and other three subbands are described in detail in the next sub-section.

Table 4.1: Parameters for four-band architecture

	Baseband	2nd Subband	3rd Subband	4th Subband
Modulation	QPSK	16-QAM	16-QAM	16-QAM
Upsampling factor	16	32	32	32
Symbol rate (Ksps)	27.125	13.57	13.57	13.57
Bit rate (Kbps)	54.25	54.25	54.25	54.25
Roll-off factor	0.4	0.4	0.4	0.4
Mixer frequency (KHz)	0	33.91	54.25	74.6
Bandwidth (KHz)	37.98	18.99	18.99	18.99
Sampling rate (Ksps)	434			
Total bandwidth (KHz)	84.088			

4.2.1 Baseband filter

The upsampling factor of the baseband is 16, to achieve a 434 kHz sampling rate. A roll-off factor for the filter is set to not only reduce ISI, but also to ensure that the excess bandwidth is moderate. The designed filter has the characteristic of finite impulse response (FIR) with a linear phase if possible. The filter order is designed as 128 and the cutoff frequency of the baseband designed is 13.57 KHz. The price paid to mitigate ISI with a raised cosine filter is the excess bandwidth according to Eq. (2.4). The baseband bandwidth including filtering is 18.99 KHz.

The frequency response of the baseband filter is illustrated in Fig. 4.5. The corresponding impulse response is shown in Fig. 4.6. In all, with the analysis of the frequency response

and impulse response, the filter designed for the baseband is appropriate to achieve a balance between ISI and excess bandwidth. The tail shown in the impulse response is relatively small, so that ISI is mild even if there is a timing error.

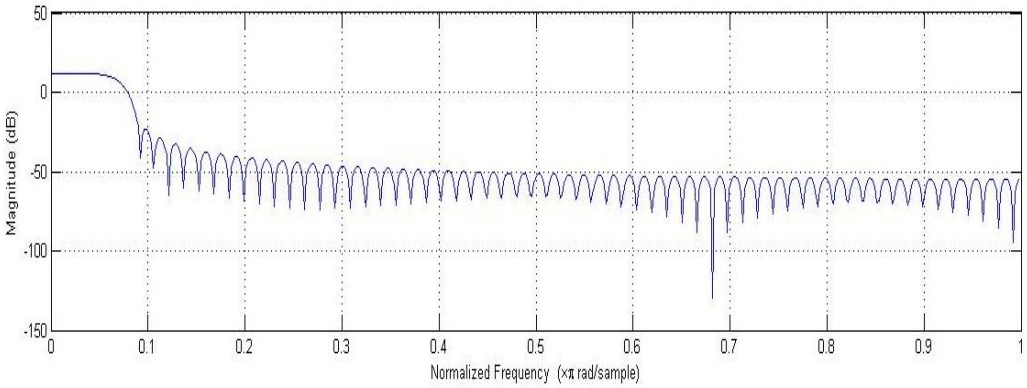

Figure 4.5: Frequency response of the baseband filter.

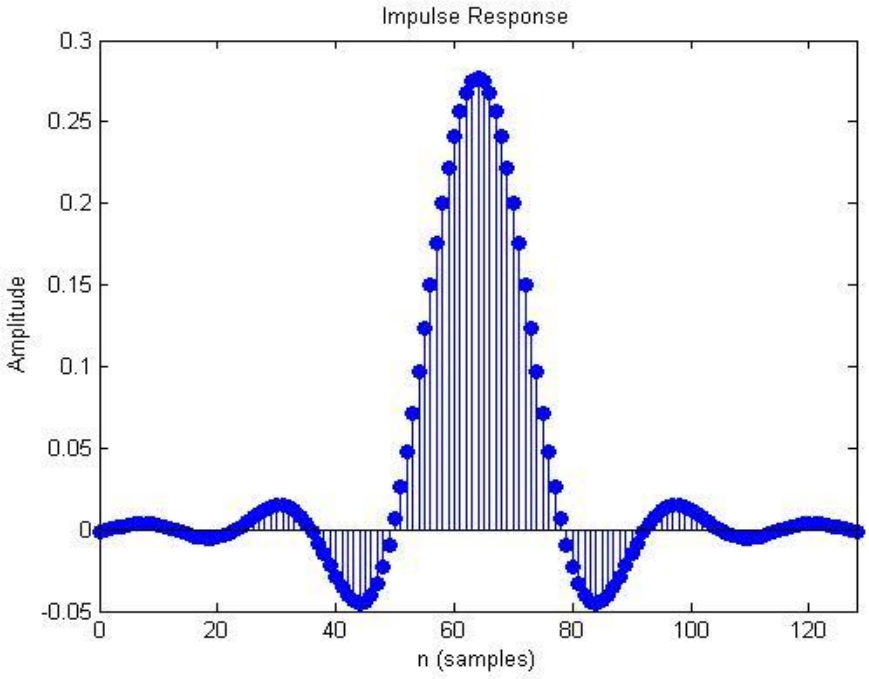

Figure 4.6: Impulse response of the baseband filter.

4.2.2 Filters for the Other Three Subbands

In order to minimize the total bandwidth of the system, one essential requirement is the

implementation of the three subbands in a narrow frequency range. The data rate for each band is 4 Ksps as listed in Table 4.1. Each subband employs 16-QAM, which results in a maximum spectral efficiency of 4bps/ Hz. The selection of 16-QAM not only makes sure the symbol rate in each subband is high, but also maintains the ISI-resistant ability. Hence the symbol rate for each subband is 13.57 KHz.

The only difference of the filter design between the higher-frequency subband and the baseband is the cutoff frequency. Due to higher-order modulation, the narrower frequency range is defined for higher spectral efficiency. Thus, the bandwidth for the subband is reduced by half equaling 13.57 KHz. The Matlab statements 'rcosine' and 'rcosflt' are also available to design this specific raised cosine filter, with the Fc (cutoff frequency) set at 13.57 KHz. The roll-off factor and filter order settings are same as the baseband filter case. The frequency response of filter for the baseband filter for other three subbands is presented in Fig. 4.7 and the impulse response of it is depicted in Fig. 4.8.

Fig. 4.7 shows the flat frequency response of channel for the frequency range of the channel from -9.50 KHz to 9.50 KHz ($0.04375 \times \pi$ rad/sample). The extra bandwidth is equal to 0.4 times the original one. Thus, the bandwidth of the passband is extended by 9.50 KHz. The maximal amplitude of the tail reads from Fig. 4.8 is -0.01 which is relatively smaller than the peak value to limit the influence of ISI.

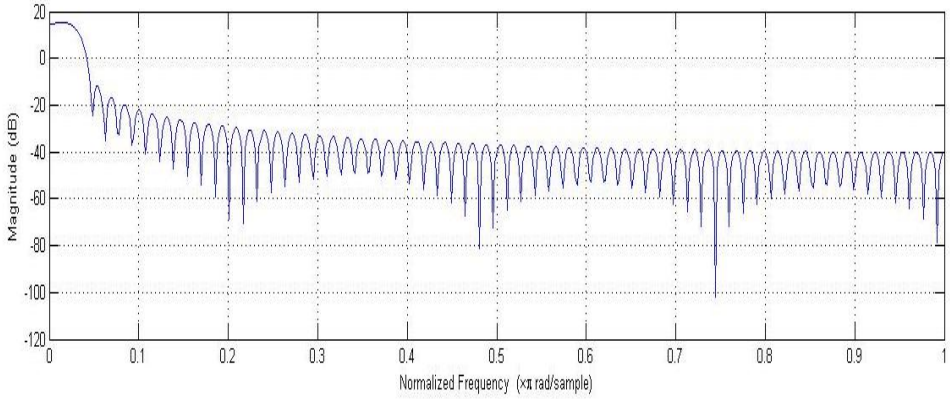

Figure 4.7: Frequency response of the baseband filter for the other three subbands.

All subbands occupy an equal bandwidth because they transmit at the same symbol rate with 16-QAM. Data symbols for each subband are filtered before the mixer, and then they will be modulated into the corresponding high frequency ranges.

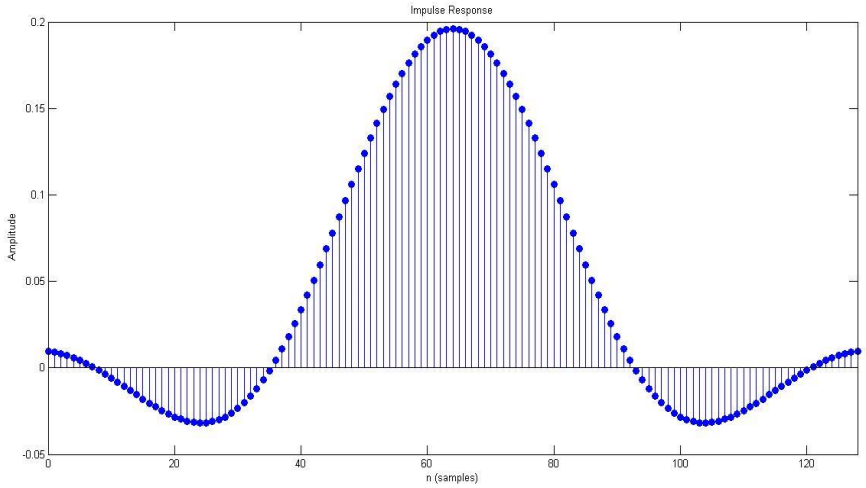

Figure 4.8: Impulse response of the baseband filter for the other three subbands.

4.3 BER Simulation Result for the Four-band Architecture

The goal is to achieve a total data rate of 217 Kbps with 4 subbands, and each of the bands

carries 54.25 Kbps. The band occupying the lowest frequency range has a symbol rate of 27.125 Ksps. The symbol rate of the other three subbands with 16-QAM is 13.57 KHz.

A total of $1.2 \cdot 10^5$ bits are simulated, and an average BER of 0.99% is observed. Note that all errors are detected in the 4th band where 20dB attenuation occurs. The baseband, the 2nd and the 3rd subbands are free of bit error. The 4th subband, where 1187 errors are detected among 30,000 bits transmitted, has a BER of 3.96%.

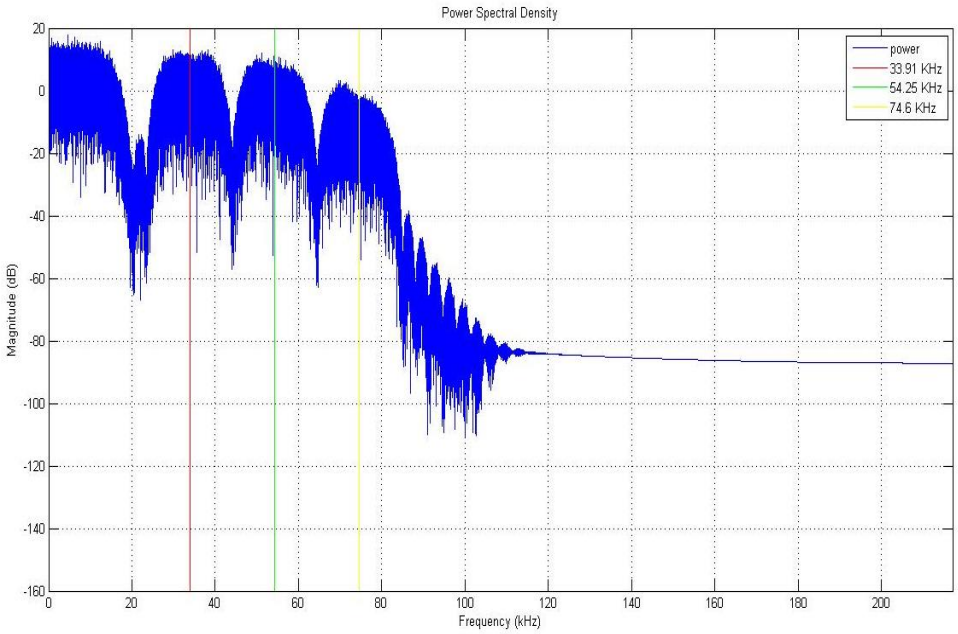

Figure 4.9: One-sided power spectrum of four-band architecture before channel.

The one-sided power spectrum of the transmitted signals before the channel with the four-band architecture is depicted in Fig. 4.9. From the picture, it is obvious that the powers of the baseband (from 0 Hz to 18.99 KHz in Fig. 4.10), the 2nd subband (center frequency at 33.91 KHz) and the 3rd (center frequency at 54.25 KHz) and 4th subband (center frequency at 74.60 KHz) are approximately at the same level.

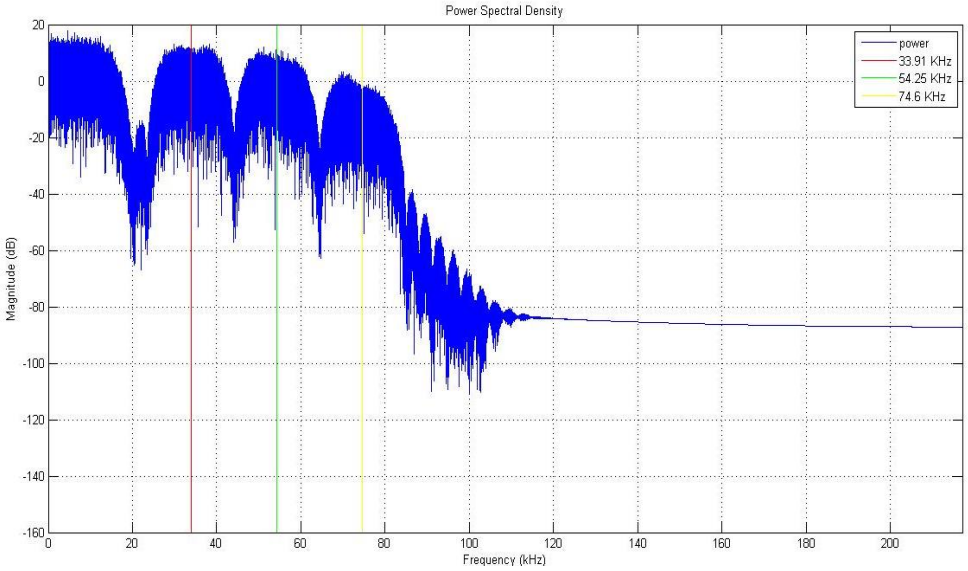

Figure 4.10: One-sided power spectrum of four-band architecture after channel.

Fig. 4.10 shows the power spectrum of signal at the output of the channel. It shows how the transmitted signal is affected by the maximally flat filter and the actual short telephone line, when compared with the result of Fig. 4.9. The power of the signal at the output of the baseband and 2nd subband did not change much after the channel, but the signals in the 3rd and 4th subbands are significantly attenuated by the channel. The power of the 3rd subband had shrunk by about 3 dB. The signal at the left edge of the 3rd subband is a few dB higher than that at the right edge. The 4th subband is the worst; the frequency component at the right edge is attenuated by 11dB more than at the right edge. This causes severe ISI even though the filters are designed to have zero ISI.

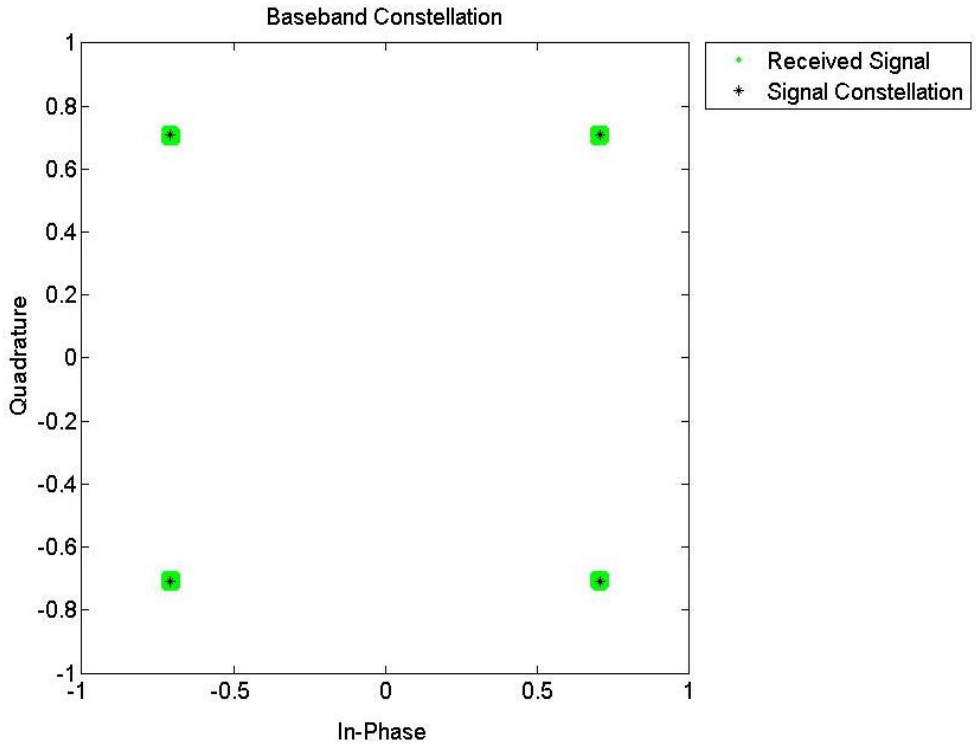

Figure 4.11: Received signal and constellation of the baseband.

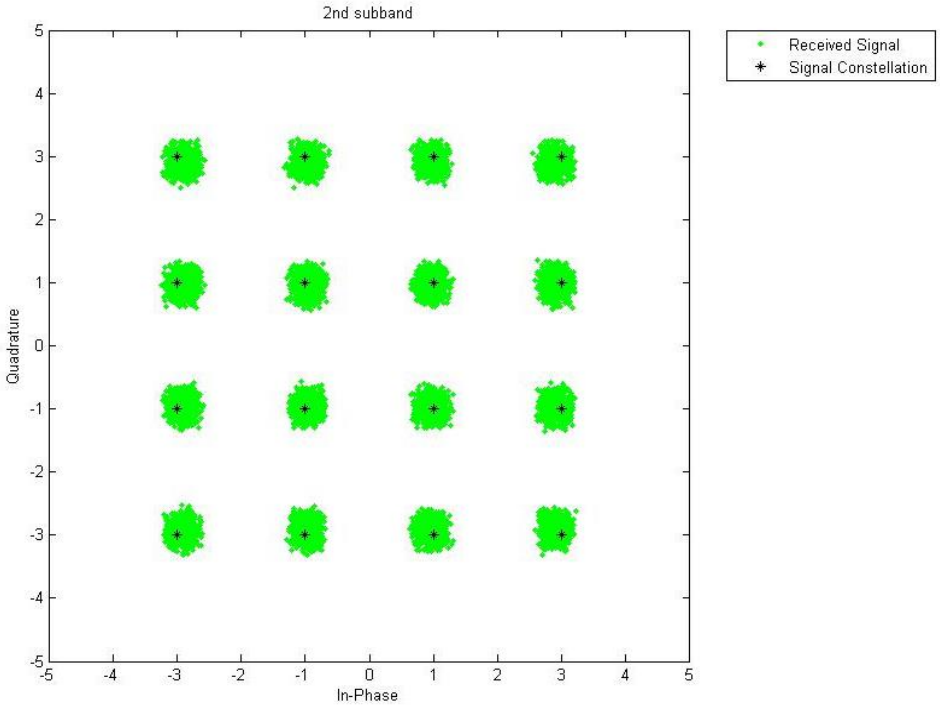

Figure 4.12: Received signal and constellation of the 2nd subband.

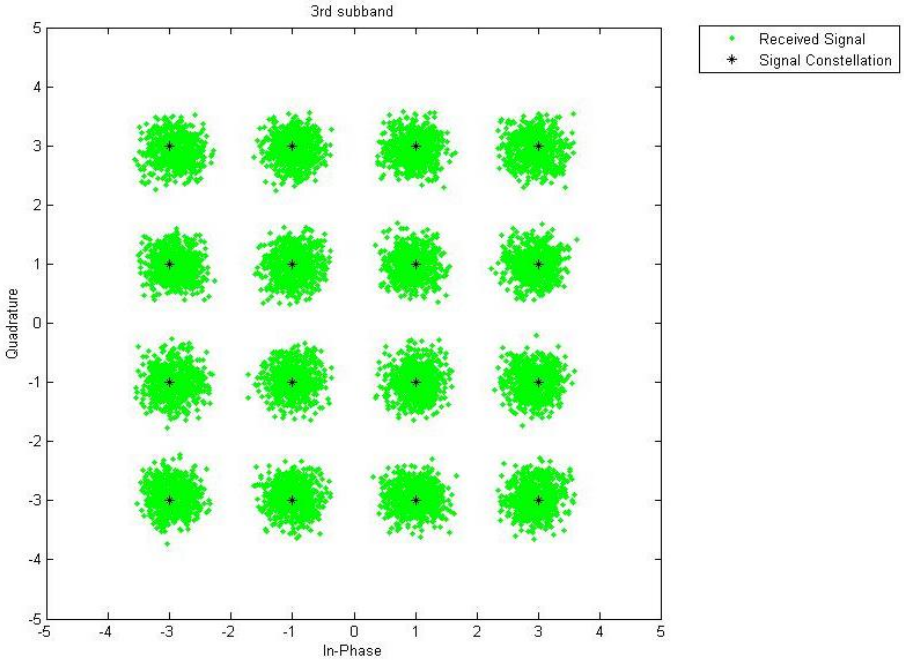

Figure 4.13: Received signal and constellation of the 3rd subband.

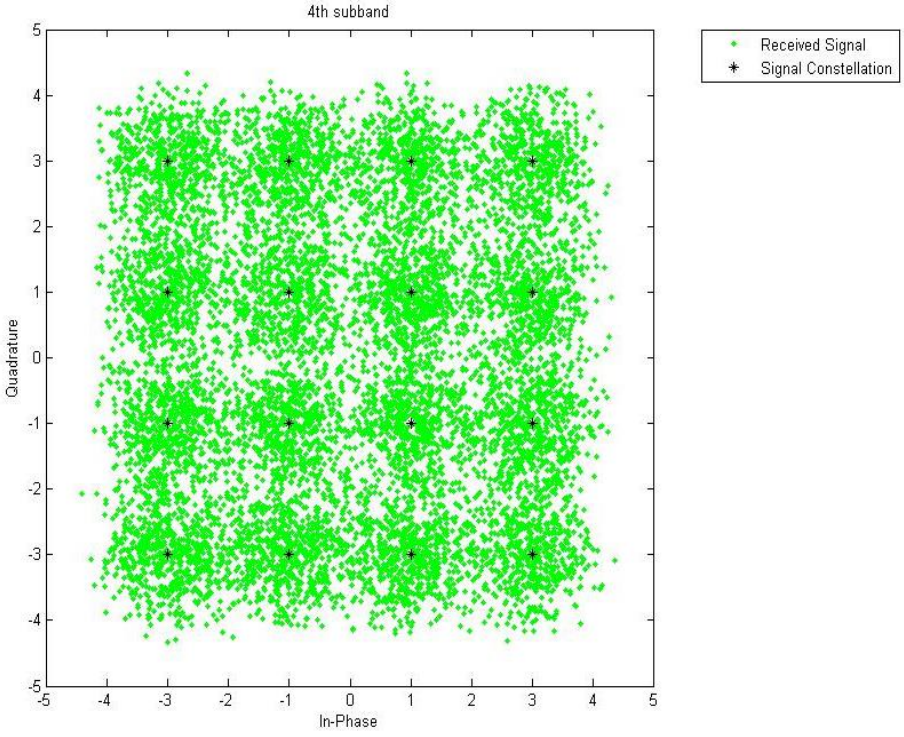

Figure 4.14: Received signals and constellation of the 4th subband.

From Fig. 4.11, the transmission in the baseband clearly shows no or ISI. Note that Gaussian noise had not been included in the simulation because of the strong signal and thus noise is negligible. The received signal centered closely at the four constellation points $\{\sqrt{2}/2,\sqrt{2}/2\}$, $\{-\sqrt{2}/2,\sqrt{2}/2\}$, $\{\sqrt{2}/2,-\sqrt{2}/2\}$, $\{-\sqrt{2}/2,-\sqrt{2}/2\}$ with QPSK which demonstrates that ISI in the baseband is negligible

The received signals shown in Fig. 4.12, Fig. 4.13 and Fig. 4.14 present a gradually increased level of ISI. Fig. 4.12 shows that the received signal at the 2nd subband, which covers the frequency range from 24.41 KHz to 43.44 KHz. From this figure, the received symbols are still centered closely within small area around the 16 constellation points. Thus, ISI in the 2nd subband is also not severe, as for the baseband.

ISI in the 3rd subband is worse, but still acceptable, from Fig. 4.13. The wide scatting of received signal for the 4th subband shows severe ISI. One approach to mitigate this ISI is to apply a lower-order modulation like BPSK, 4-QAM to minimize ISI effects. Another approach is to apply a pre-distortion filter to equalize the channel.

5. FPGA Implementation

The FPGA implementation is not simple, because it involves the filter design in Matlab, Verilog coding, Modelsim simulation and other lab experiments. The overall four-band architecture in FPGA simulation requires generation of a pseudo-random code in addition to the modulation, the mixer, and the filter design in Matlab. All blocks, including the pseudo-random code, filter and modulation, are implemented with Verilog, simulated in Modelsim, and synthesized by Xilinx XC5VLX110T FPGA.

5.1 Pseudo-random Code

The desired implementation should be done under conditions that most emulate the real environment. However, in actual implementation, the circuit uses the combinational logic to achieve the logical function, which means that all functions are realized with logic gates. In contrast to the random behaviors of signals in the real world, which is unpredictable, the simulated randomness is not truly random. Thus, the pseudo-random code, which is an algorithm for generating a sequence of numbers that approximates the properties of random numbers, is important in practice to generate signals that emulate the real environment.

Assuming the code has 4 bits, the algorithm applied for generating a pseudo-random code is:

$$y = 9 \cdot x + 3 \tag{5.1}$$

Thus, the bit sequence is in loops, as shown in Table 5.1

Table 5.1: Look up table between binary and decimal

Bit(binary)	Number(decimal)	Bit(binary)	Number(decimal)
0000	0	1011	11
0011	3	0110	6
1110	14	1001	9
0001	1	0100	4
1100	12	0111	7
1111	15	0010	2
1010	10	0101	5
1101	13	0000	0
1000	8	0011	3

The two columns on the right of the table are continued from the left two. In the case of the symbol exceeding 16, for example one symbol is 3 thus the next one should be 30 (11110) in decimal, the last four bits (1110) are accepted as the next symbol. Each loop containing 16 symbols starts from 0000 and ends at 0101, which means that every 64 bits constitutes one loop.

5.2 Modulation

The modulation methods used in this design all require in phase and quadrature components.. In order to reduce BER, gray code mapping between bits and symbols is used,

which ensures that the neighbor symbols have only one bit difference. For example in 16-QAM case:

Fig. 5.1 presents the symbol position, following gray code mapping, and shows that the neighboring symbols have only one bit difference. From Table 5.2, the amplitude of both in-phase and quadrature components are assigned according to gray code mapping.

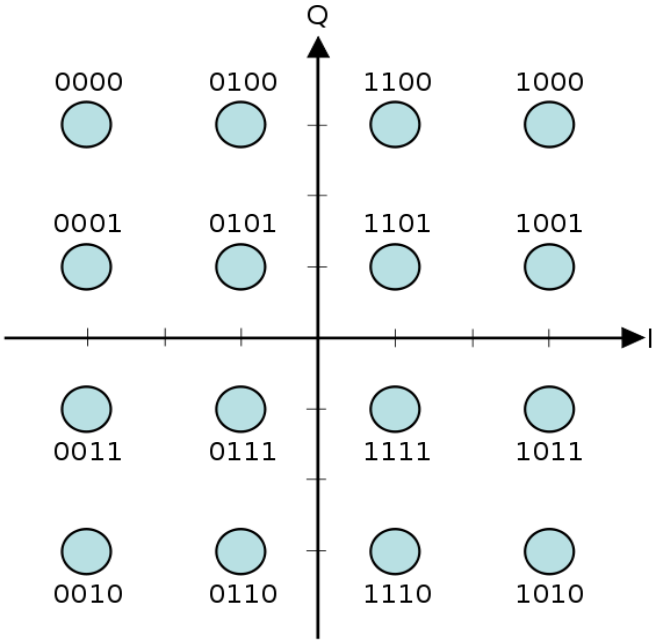

Figure 5.1: Constellation of 16-QAM.

Table 5.2: In-phase and quadrature amplitude components for 16-QAM modulation

Symbol	In phase	Quadrature	Symbol	In phase	Quadrature
0000	-3	3	1000	3	3
0001	-3	1	1001	3	1
0010	-3	-3	1010	3	-3
0011	-3	-1	1011	3	-1
0100	-1	3	1100	1	3

0101	-1	1	1101	1	1
0110	-1	-3	1110	1	3
0111	-1	-1	1111	1	-1

QPSK is similar to QAM as shown in Table 5.3:

Table 5.3: In-phase and quadrature amplitude for QPSK modulation

Symbol	In phase	Quadrature
00	1	1
01	-1	1
10	1	-1
11	-1	-1

5.3 Filter Implementation

This implementation includes two types of FIR filter which are raised cosine filter for pulse shaping and a maximally flat filter to emulate the channel attenuation. The design and analysis process has already been discussed in Chapter 4. This section focuses on the implementation in FPGA.

Filter process is the process of convolution,. First, the coefficients of the designed filter can be found from the FDATool in Matlab. Then decimal numbers are converted into binary numbers. For example, if the decimal signal is 0.25, then the binary format should be 0.01. A problem that occurs during conversion is that hardware doesn't recognize the radix point. The solution refers to the fact that the radix point can be presented as a float point, that is, it

can be placed anywhere relative to the significant digits of the number, like the floating point of 0.25 being regarded as 1. The 2's compliment has the advantage for arithmetic operations of addition, subtraction and multiplication for unsigned binary number, especially efficient for negative numbers.

Table 5.4: 2's complementation for filter coefficient

Filter Coefficient Number	Decimal	Binary (Signed)	2's complement (4 bits)
1-15	Less than 0.0313	0	0000
16	-0.0335	1.1111	1111
17	-0.0252	0.0000	0000
18	0.0339	0.0001	0001
19	0.1395	0.0010	0010
20	0.2458	0.0100	0100
21	0.2908	0.0101	0101

Table 5.4 shows the floating point in 4bits of the filter coefficients after 2's complements. Symmetry is one of the properties of the filter coefficient, assuming *N+1* parameters exist (interval *[0, N-1]*) so that the *M*th parameter is equal to $(N - M)$th.

The mathematical operation of filtering is convolution and can be expressed as:

$$f(m) * g(m) = \sum_{m=0}^{N-1} f(m) \cdot g(n-m) \tag{5.2}$$

where $f(m)$ is the input signal and $g(m)$ represents filter coefficients. Due to the symmetric structure of the filter impulse response, Eq. (5.2) can be rewritten as:

$$f(m)*g(m)=\sum_{m=0}^{\frac{n}{2}-1}\left[f(m)+f(n-m)\right]g(n-m)+f(\frac{n}{2})g(\frac{n}{2}) \quad (5.3)$$

From Eq.(5.3), the multiplication operation has been reduced by a half compared to Eq. (5.2), which saves lots of DSP resources in FPGA.

Digital signal processing is major source of power consumption, since it requires multiplication and addition in one clock period. The symmetry property of the filter coefficients has been utilized to minimized required computational resources, but the addition of hundreds of registers in one path is still significant for implementation in FPGA. The approach called pipeline can help to solve this problem by splitting one multiplication into several clocks. It might require more registers, but FPGA provides far more than enough and eliminated the number of other sources like LUT. The main route has simplified so that the delay of the route is reduced with higher frequency is allowed for design. For example, if FPGA does a 6 input addition operation in serial, it should look like Fig. 5.2.

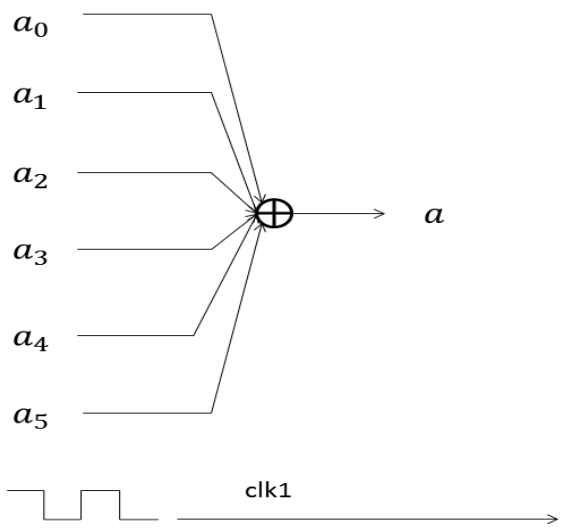

Figure 5.2: Serial operation of addition.

Only one register is required, which can be described as a synchronous digital circuit with 6 inputs. The FPGA has to assemble huge amounts of logic gates and a hierarchy of reconfigurable interconnects that allow the blocks to be "wired together" to implement the addition operation.

Fig. 5.3 shows the parallel processing with pipeline approach. It is obvious that each of the 3 clock cycles generate one output signal. This structure applies 3 more registers than the serial case, but the synchronous logic gate block only connects with 2 inputs which means less LUT and other source utilization. The route does not require solving a high demand addition operation in one clock cycle. Although the output will be delayed by 2 clock cycles, all registers can be operated simultaneously and productively because they are arrayed as pipeline in synchronization. It has higher frequency allowance, less source utilization, and less power consumption with pipeline structure for FPGA when dealing with complex digital signal processing.

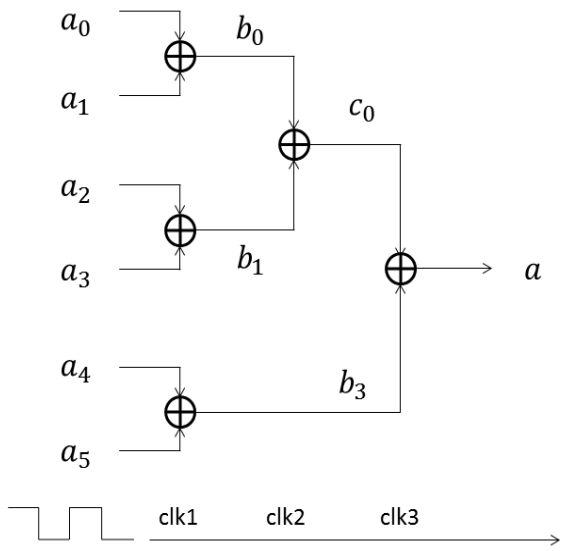

Figure 5.3: Parallel operation of addition.

5.4 User Constraints

5.4.1 Timing Constraints

Referring to the synthesis timing report, the details about the default period analysis for Clock 'cclk_OBUF1' are listed for timing constraints. The delay for one data path within 6 gates consumes at least 8.794 ns, which depicts the maximum clock frequency up to 113.714 KHz. The minimum input arrival time before clock is 2.994ns and the output required time after clock is 2.783ns.

There are several selective crystal oscillators integrated with the development board available for users. Due to the restriction of ADC and DAC, the symbol rate supported for design has to be limited within 1Msps, which means the achieved maximum frequency is 16MHz (16bit/symbol). One available Clock pin IO_L6P_GC3 connected with SMA_DIFF_CLK_OUT_P, capable of the frequency of 250MHz. In order to fit the clock requirement of within 16MHz, one frequency divider is manually created by Verilog so as to slow the clock period. After being divided by 18, this clock frequency meets the requirement and drives all the logic blocks and flip-flops in 13.88 MHz.

5.4.2 I/O Pin Planning

The UCF file is capable of editing both the timing constraints and the I/O pin assignment. Reference to ‘ml50x schematics’ that illustrates wire connection to FPGA with other chips in allows, each input and output pin of module to be assigned to the exact I/O of FPGA for further implementation. As mentioned in Chapter 2, the input signal ‘clk’ is assigned in LOC ‘H14’ which represents the IO_L6P_GC3 according to the schematic, connected with

SMA_DIFF_CLK_OUT_P from the crystal oscillator. The development board left sufficient open headers to connect with the DAC and the ADC for control, power supply and data input/output signal. Browsed the schematic diagram, J4 and J6 header's I/O can be utilized, each header providing 5 pins to connect with the headers of the ADC and the DAC.

5.5 Hardware Implementation

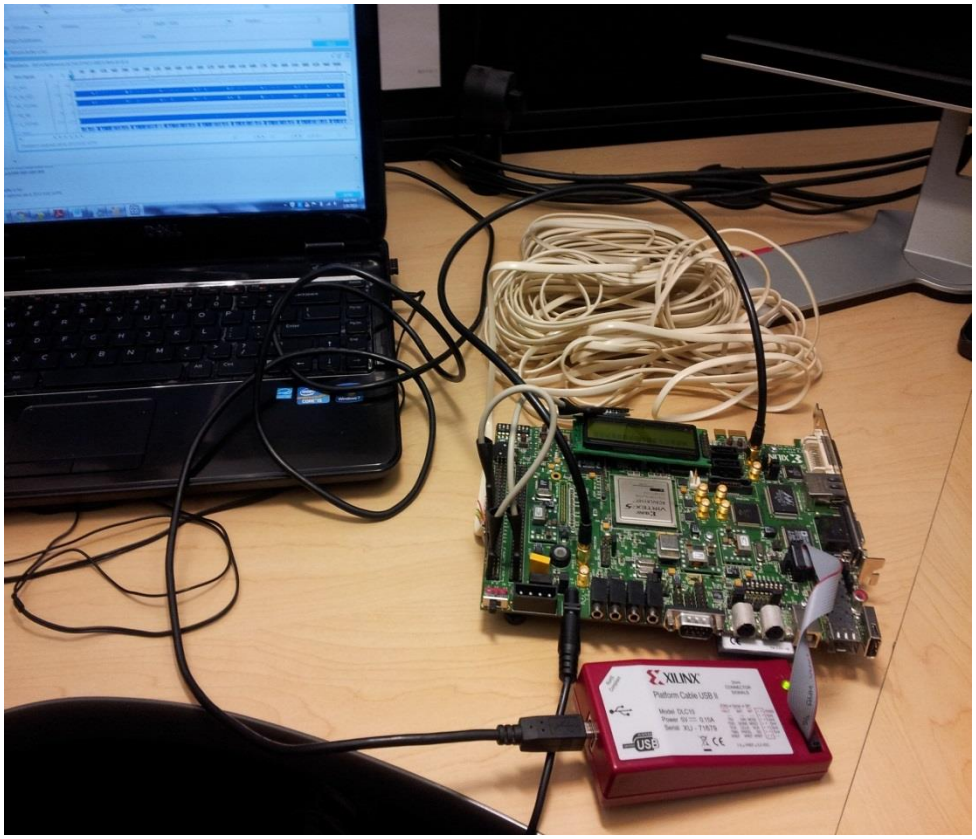

Figure 5.4: Hardware implementation.

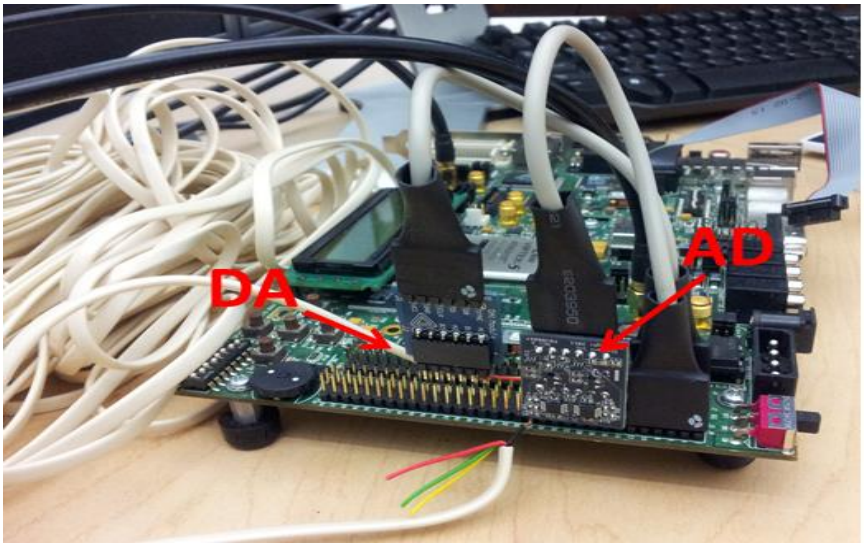

Figure 5.5: The connection of DAC and ADC.

Fig. 5.4 illustrates total hardware implementation of the design including the telephone line, the ADC, the DAC, JTAG and the computer. The telephone line is connected between the ADC and the DAC. JTAG works as a bridge for configuring and programming design from the computer to FPGA. The computer is used to program and analyze the result from ChipScope for debugging. Fig. 5.5 presents the detail of the ADC and the DAC connection. The pin cable connects with the chip and the J7 header which provides the control, power and input signal for the DAC. Another output pin of the DAC is tied with telephone line. Due to the control and power pins being designed in the same side as the input pins, the telephone line is tied with the header and the input pins together. The decoded signal is transmitted through the cable to the J4 header, where FPGA can interpret for further demodulation.

5.6.1 The Single-band Architecture Implementation Result

The Clock frequency applied at 13.88 MHz and each sampling rate consumed 16 clock cycles. The bandwidth occupied 151.9 KHz for delivering the bit rate as 217 Kbps. The

symbol rate equals the data rate because each symbol contains one bit for BPSK. Fig. 5.6 shows the screenshot of the baseband design from ChipScope which captures the internal signals and displays them as any numeral base. The signal labeled as ‘Baseband’ exhibits the received symbol of the single-band. The rectangle encloses one sequence from symbol ‘3’ to symbol ‘0’ and the circle is highlighted where BER occurs. The signal should be in binary for BPSK modulation, but it arrayed as hexadecimal. This was due to the input signal following the pseudo-random code algorithm mentioned in previous section and the errors are apparent in 4 bit/symbol format.

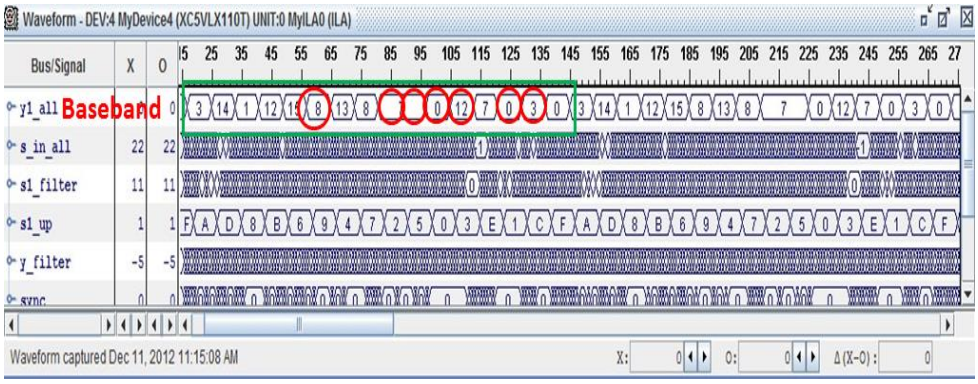

Figure 5.6: ChipScope detection result for the single-band architecture.

Table 5.5: Comparison between received and transmitted symbol of the single-band architecture

Transmitted		Received		Transmitted		Received	
Hexadecimal	Binary	Hexadecimal	Binary	Hexadecimal	Binary	Hexadecimal	Binary
3	0011	3	0011	11	1011	7	0111
14	1100	14	1100	6	0110	7	0111
1	0001	1	0001	9	1001	0	0000
12	1100	12	1100	4	0100	12	1100

15	1111	15	1111	7	0111	7	0111
10	1010	8	1000	2	0010	0	0000
13	1101	13	1101	5	0101	3	0011
8	1000	8	1000	0	0000	0	0000

The result of BER for Single-band implemented in FPGA is 15.625%. Compared with two lists of symbols from Table 5.5, it is observed that 7 pairs do not match which means 7 symbol errors occurred among each sequence. Three of these errors found a 2 bit difference and others detected a 1 bit difference. In conclusion, 10 bit errors occurred in 64 bits. The error always happens in the same position in each sequence because the pseudo-random code has the property of repetition and is not complex enough to simulate the real random. The result of FPGA implementation is better than 16.74% BER of Matlab simulation, which is due to pseudo-random code as input. In summary, the telephone line and the manually made maximally flat filter approximately simulate the feature of the channel response as 130 dB attenuation at 151.9 KHz bandwidth from the result of FPGA implementation in practice.

The main device utilizations are listed in the following table. Due to the simple architecture of single-band, only a small slot of the device is utilized for implementation.

Table 5.6: FPGA device utilization for the single-band architecture

Virtex-5 XC5VLX110T				
Device Type	Slice LUTs	Slice flip-flop	DSP48Es	Max Clock Frequency
Number of Device	659	937	2	176.897 MHz
Utilization	1%	1%	3%	--

5.7 The Four-band Architecture Implementation Result

The target of four-band architecture is to deliver a singal of 217 Kbps, where each band is splited equally into 54.25 Kbps to achieve the same total bit rate as the single-band approach. The bandwidth of four-band architecture occupies 84.088 KHz, delivering a symbol rate of 434 Ksps. The baseband dominates a 37.98 KHz frequency range and the subband requires 18.99 KHz. The clock cycle follows the design of single-band to ensure the comparsion under the same test environment.

The FPGA implemention result is exhibited in the sceenshot of Chipscope in Fig. 5.7. The demodulated signal is labeled as ‘baseband’, ‘2nd subband’, ‘3rd subband’ and ‘4th subband’, with their symbols marked as a rectangle. The sequence of pseudo-random symbol for baseband is in the order of ‘3’, ‘2’, ‘1’,‘0’ and the other subband follows the same pseudo-random algorithm as single-band design.

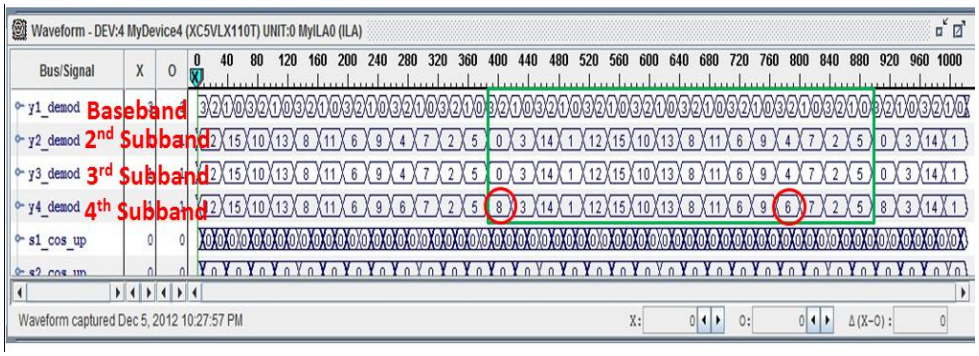

Figure 5.7: ChipScope detection result for the four-band architecture.

Table 5.7: Comparison between received and transmitted symbol of the four-band architecture

Transmitted		Received		Transimtted		Received	
Hexadecimal	Binary	Hexademical	Binary	Hexadecimal	Binary	Hexadecimal	Binary
3	0011	3	0011	11	1011	11	1011
14	1100	14	1100	6	0110	6	0110
1	0001	1	0001	9	1001	9	1001
12	1100	12	1100	4	0100	6	0110
15	1111	15	1111	7	0111	7	0111
10	1010	10	1010	2	0010	2	0010
13	1101	13	1101	5	0101	5	0101
8	1000	8	1000	0	0000	8	0100

The implemantion result of four-band architecture is 0.78% BER, which is close to the Matlab simulation result of 0.99%. There were only 2 errors detected in the total received signal in one sequence where all found in the 4th subband. Table 5.7 lists the received symbols from the 4th subband so as to detect where error happens.These incorrect symbols

have 1 bit difference with origin, so the BER for the 4th subband is 3.125%. From the implementation result, it is obvious that the baseband, the 2nd and the 3rd subband are free of bit error. The fourbband architecture analysis desrcibes how ISI induced from the channel frequency response influences starting at the 3rd subband, but the successful raised cosine filter has resistce to the effect of ISI and results in zero bit error. The 4th subband detected errors because it had sufftered about 20 dB attenuation. In summary, the pratical result of 0.78% BER of FPGA implemation approaches the theortical BER as 0.99%. Analyzing the result, the reason why practical is better than theotrical because the loop of one pseudo-random code sequence is too narrow to simulate the real random.

The device utilzation is listed in Table 5.8, where the assumption is that more devices are required to be involved for four-band architecture to be proved. It is because of the compliexity of the logic synthesis of the architecure, including two modulations and two styles of filters. From the table, the DSP48Es is used for digital signal processing which occupies 12 of 64 for synthesis.

Table 5.8: FPGA device utilization for the four-band architecture

Virtex-5 XC5VLX110T				
Device Type	Slice LUTs	Slice flip-flop	DSP48Es	Max Clock Frequency
Number of Device	12403	18262	12	113.714 MHz
Utilization	17%	26%	18%	--

6. Conclusion and Future Work

6.1 Conclusion

This thesis implements and tests two transmission architectures over severe ISI channels and compares them in terms of BER. The four-band scheme seems to work better with negligible ISI due to appropriate raised cosine filters and relatively flat channel frequency response in each subband. The channel characteristics are analyzed extensively for the architecture design. Major efforts are spent on defining the proper frequency range for each band and guard bands, which are all simulated in Matlab and implemented in FPGA.

The four-band architecture, which results in a 0.78% BER, shows many benefits compared to the single-band solution. The flat channel frequency response within each subband ensures signaling without an equalizer. The higher-order modulation (16-QAM) for each subband increases the rate, enabling the four-band architecture to require a total bandwidth of 84.09 KHz only, which is significantly smaller than 151.09 KHz required with the single-band approach. From FPGA implementation results, the BER of the four-band approach is dramatically lower than the single-band approach of 15.625%, which demonstrates that it is more suitable to transmit at higher data rates than the single-band approach. The mixer, carrier tracking (frequency and phase) and filters for each sub-band are also implemented in FPGA.

6.2 Future Work

Channel coding is classified into two categories: error detection codes and error correction codes that enable reliable delivery of digital signals through unreliable channels. Many channels are vulnerable to the noise or ISI, causing detection error.

Future work includes optimizing error correction codes that are simple and of high rate for severe ISI channels. This includes implementation in FPGA and comparison with the case without coding in terms of BER, bit rate, and bandwidth.

[1] Christian Weber, “Multiband Architecture for a 25 Gbps High Speed SerDes on a Backplane Channel: Analysis, Simulation and Speciation”, MS thesis, Offenburg University of applied Sciences, Germany and Oregon State University, USA, Sep. 2009.

[2] G. D. Forney, Jr., “Maximum-Likelihood sequence estimation of digital sequences in the presence of intersymbol interference”, *IEEE Trans. Information Theory*, vol. IT-18, no. 3, pp. 362-378, May 1972.

[3] J. O. Scanlan. “Pulses satisfying the Nyquist criterion” *ELECTRONICS LETTERS*, Vol. 28, no. 1, pp.50-52, 2nd January, 1992

[4] Nyquist, Harry. "Certain factors affecting telegraph speed", *Bell System Technical Journal*, 3, 324–346, 1924

[5] Wikipedia. Phase-shift keying http://en.wikipedia.org/wiki/Phase-shift_keying

[6] Chrisikos, G., "Analysis of 16-QAM over a nonlinear channel," *Personal, Indoor and Mobile Radio Communications, 1998. The Ninth IEEE International Symposium on* , vol.3, no., pp.1325,1329 vol.3, 8-11 Sep 1998

[7] Wikipedia Xilinx http://en.wikipedia.org/wiki/Virtex_(FPGA)#Virtex_family

[8] Virtex5 FPGA Data Sheets

[9] Xilinx Timing Constraints User Guide ,UG612 (v1.0.0) December 9, 2008

[10] Wikipedia, Upsampling http://en.wikipedia.org/wiki/Upsampling

[11] Wikipedia, Analog-to-digital converter http://en.wikipedia.org/wiki/Analog-to-digital _converter

[12]Wikipedia, Digital-to-analog converter http://en.wikipedia.org/wiki/ Digital-to-analog _converter

[13] Digilent PmodAD1™ Analog To Digital Module Converter Board Reference Manual

[14] AD7476A Data Sheet http://www.analog.com/static/imported-files/ data_sheets /AD7476A_7477A_7478A.pdf

[15] AD7303 Data Sheet http://www.analog.com/static/imported-files/data_sheets/AD730 3.pdf

[16] R. W. Hamming, Error Detecting and Error Correcting Codes, *Bell System Tech*. J. 29 pp. 147-160 (1950).

[17] Elias, P., "Error-free Coding," *IRE Trans. on Information Theory*, vol. 4, no. 4, pp.29,37, Sep. 1954