The Islamic University of Gaza

Electrical Engineering Department

Design and Optimization Loop Filters in Fixed WiMAX PLL using LMI Method

By

Fady Fadel EL-Batta

Supervisors

Dr. Hatem ELaydi & Dr. Hala J. El-Khozondar

Submitted in Partial Fulfillment for the Master Degree in Electrical Engineering

November 2009

Dedicated to

My Father, Mother, brothers and sister.

Abstract

Achieving an optimal design of phase-locked loop (PLL) is a major challenge in WiMAX technology in order to improve system behavior against noise and to enhance Quality of Service (QoS). A new loop filter design method for PLL is introduced taking into consideration various design objectives such as small settling time, small overshoot and meeting Fixed WiMAX requirements. Optimizing conflicting objectives is accomplished using semi-Definite programming (especially Linear Matrix Inequality (LMI)) in conjunction with appropriate adjustment of certain design parameters. Infinite Impulse Response (IIR) and Finite Impulse Response (FIR), Digital filters are designed by Semi-Definite Programming (SDP) using SeDuMi (self-dual minimization) toolbox software.

Design efficiency and performance of the proposed method are illustrated by simulations and comparisons to other design methods. Simulations showed that the IIR digital low pass filter which was designed by SDP is better than the IIR digital low pass filter which Al-Quqa "in [32]" designed by Linear Programming (LP). Simulations show that the FIR digital low pass which was designed by SDP using SeDuMi software is better than the FIR digital low pass filter which was designed in [32] by SDP using CVX (convex optimization tool) software. The minimum-length FIR filter algorithm was used to proof that the order of the FIR filter which was designed using SDP method and simulated using toolbox software (SeDuMi) is optimal for our design specifications.

Acknowledgements

Praise be to Allah, the Almighty for having guided me at every stage of my life. I have been fortunate to have the opportunity to work under the supervision of Dr. Hatem Elaydi and Dr. Hala J. El-Khozondar. Their guidance, valuable suggestions and support throughout all stages of my thesis are highly appreciated. I would also like to place on record my appreciation for the cooperation and guidance by my committee members, Dr. Basil Hamad and Dr. Ammar Abu Hudrouss.

I recall with deep gratitude and respect the debt I owe to my mother, brothers and sister for their unfathomable love, prays, support and encouragement throughout my studies. I would also like to thank my best friends Ayman Al-Quqa and Wessam Al-Moamar for supporting me during my thesis. May Allah bless them all.

Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

Filter design is one of the most important problems in Signal Processing. Many techniques are available to design analog as well as digital filters. However research in the area of digital filter design is still active to find new kinds of filters with better performance and the methods for designing them, or to provide more reliable, efficient and convenient design algorithms. We will design IIR and FIR, Digital filters by SDP using SeDuMi toolbox software. The designed digital filter will be used as loop filter for PLL, which the main component in the frequency synthesizer. The frequency synthesizer is electronic device used in Fixed WiMAX communication system.

Fixed WiMAX is the fourth generation of communication systems. It was used in Gaza strip as communication technology to building the computer network between some schools. We will employ multi-objective control technique to deal with the various design objectives such as small settling time, small overshoot and meeting Fixed WiMAX requirements. Trade-off among the conflicting objectives will be made via using SDP in conjunction with appropriate adjustment of certain design parameters. Numerical simulation on nonlinear PLL model and comparisons to other design methods will be performed which demonstrates the effectiveness of the proposed method.

In this chapter we introduce summary information about the background of WiMAX technology, PLL fundamentals, frequency synthesizer, LMI method, SDP problem and finally we present the problem which we want to solve it.

1.1 Background

Wireless communications networks have been rapidly developed in the past 15 years. Wireless mobile services grew from 11 million subscribers worldwide in 1990 to more than 2 billion in 2005 [1]. During the same period, the Internet grew from being a curious academic tool to having about billion users. This staggering growth of the Internet is driving demand for higher-speed Internet-access services, leading to a parallel growth in broadband adoption. In less than a decade, broadband subscription worldwide has grown from virtually zero to over 200 million [1].

A number of technologies currently exist to provide users with high-speed digital wireless connectivity; such as fourth-generation wireless system (4G). In a fourth-generation wireless system, cellular providers have the opportunity to offer data access to a wide variety of devices. The most telling example is growth of the Internet over the last 10 years [2].

The main advantages points for 4G are [3]:

- It produces a high-speed mobile wireless access with a very high data transmission speed.
- It provides an end-to-end IP solution where voice, data and streamed multimedia can be served to users on an "anytime, anywhere" basis at higher data rates than previous generations.
- It is better than third generation communication systems such as Wi-Fi because it has been improved performance, scalability, QoS, rang and security.

One of the main technologies in the fourth generation (4G), is “WiMAX.” WiMAX, the Worldwide Interoperability for Microwave Access, is a telecommunications technology aimed at providing wireless data over long distances in a variety of ways, from point-to-point links to full mobile cellular type access. It is based on the IEEE 802.16 standard, which is also called Wireless MAN [1].

WiMAX allows a user, for example, to browse the Internet on a laptop computer without physically connecting the laptop to a wall jack. The name WiMAX was created by the WiMAX Forum [4], which was formed in June 2001 to promote conformance and interoperability of the standard. The forum describes WiMAX as "a standards-based technology enabling the delivery of last mile wireless broadband access as an alternative to cable and DSL."[1].

The WiMAX family of standards addresses two types of usage models: a fixed-usage model (IEEE 802.16-2004) and a portable or mobile usage model (IEEE 802.16e-2005). The original 802.16 standard was released in 2004 and it was only capable of providing fixed wireless data services. It used frequency band from 2 GHz to 11GHz [4]. A significant part in WiMAX system is the phase-locked loop.

A phase-locked loop (PLL) is an electronic control system that generates a signal that has a fixed relation to the phase of a "reference" signal. PLL circuit responds to both the frequency and the phase of the input signals, automatically raising or lowering the frequency of a controlled oscillator until it is matched to the reference in both frequency and phase. PLL is an example of a control system using negative feedback [5].

PLL, as shown in Figure (1.1), are widely used in radio, telecommunications, computers and other electronic applications. They may generate stable frequencies, recover a signal from a noisy communication channel, or distribute clock timing pulses in digital logic designs such as microprocessors. PLL are used in WiMAX as a frequency synthesizer [5].

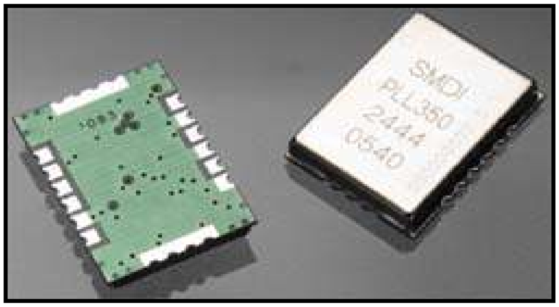

Figure 1-1: PLL Modules

A frequency synthesizer is an electronic system for generating any of a range of frequencies from a single fixed time base or oscillator to high frequencies. Since a single integrated circuit can provide a complete phase-locked-loop building block, the technique is widely used in modern electronic devices, with output frequencies from a fraction of a cycle per second up to many gigahertzes [6].

Later, I will discuss the PLL fundamentals and PLL communication application.

1.2 Phase Locked Loop Fundamentals

A Phase Locked Loop (PLL) is a feedback control circuit. A PLL is a device which causes one signal to track another one [5]. It keeps an output signal synchronizing with a reference input signal in frequency as well as in phase. More precisely, the PLL is simply a servo system, which controls the phase of its output signal in such a way that the phase error between output phase and reference phase reduces to a minimum [5]. A basic form of a PLL consists of three fundamental functional blocks namely [6]

- A Phase Detector (PD)
- A Loop Filter (LF)
- A voltage controlled oscillator (VCO)

With the circuit configuration shown in Figure 1-2

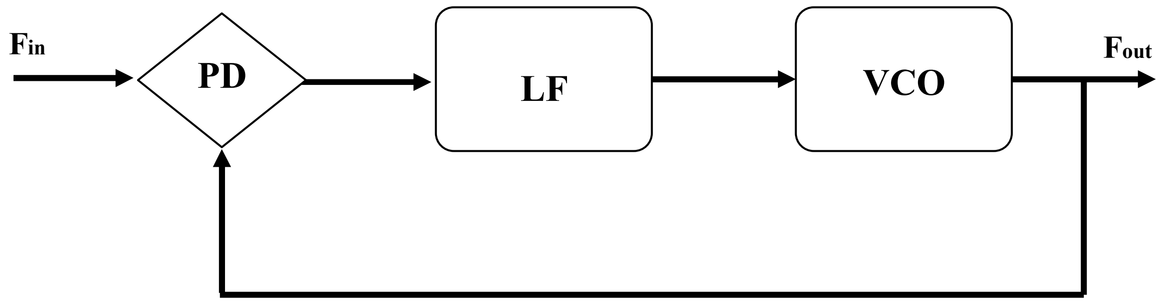

Figure 1-2: A basic PLL Block

1.2.1 Phase Detector (PD)

A phase detector (PD) is a frequency mixer or analog multiplier circuit that generates a voltage signal which represents the difference in phase between two signal inputs and it generates high and low frequencies. This output voltage passes through the loop filter and then as an input to the voltage controlled oscillator (VCO) which controls the output frequency. Due to this self correcting technique, the output signal will be in phase with the reference signal. When both signals are synchronized the PLL is said to be in lock condition. The phase error between the two signals is zero or almost zero. As long as the initial difference between the input signal and the VCO output frequency is not too big, the PLL eventually locks onto the input signal. This period of frequency acquisition, is referred as pull-in time, this can be very long or very short, depending on the bandwidth of the PLL. The bandwidth of a PLL depends on the characteristics of PD, voltage controlled oscillator and on the loop filter. Detecting phase differences is very important in many applications, such as motor control, servo mechanisms, and demodulators [6].

1.2.2 Loop Filter (LF)

The filtering operation of the error voltage (coming out from the Phase Detector) is performed by the loop filter. The output of PD consists of a dc component superimposed with an ac component. The ac part is undesired as an input to the VCO; hence a low pass filter is used to filter out the ac component. Loop filter is one of the most important functional blocks in determining the performance of the loop. Low pass filter (LPF) is a filter that passes low frequency signals but attenuates (reduces the amplitude of) signals with frequencies higher than the cutoff frequency. The actual amount of attenuation for each frequency varies from filter to filter. It is sometimes called a high-cut filter when used in audio applications. A low pass filter (LPF) smoothes out sudden changes in the control voltage [6].

1.2.3 Voltage Controlled Oscillator (VCO)

VCO is an electronic oscillator (nonlinear device) designed to be controlled in oscillation frequency by a voltage input. The frequency of oscillation is varied by the applied DC voltage, while modulating signals may also be fed into the VCO to cause frequency modulation (FM) or phase modulation (PM); a VCO with digital pulse output may similarly have its repetition rate (FSK, PSK) or pulse width modulated (PWM) [6].

The PLL, VCO, reference oscillator, and phase comparator together comprise a frequency synthesizer. Wireless equipment that uses this type of frequency control is said to be frequency-synthesized [6].

1.3 PLL Application: Frequency Synthesizer

One of the most common uses of a PLL is in Frequency synthesizers of Wireless systems. A frequency synthesizer generates a range of output frequencies from a single stable reference frequency of a crystal oscillator to very high frequencies [10]. Many applications in communication require a range of frequencies or multiplication of periodic signals. For example, in most of the FM radios, a phase locked loop frequency synthesizer technique is used to generate 101 different frequencies. Also most of the wireless transceiver design methods employ a frequency synthesizer to generate highly accurate frequencies, varying in precise steps, such as from 600 MHz to 800 MHz in steps of 200 KHz. Frequency Synthesizers are also widely used in signal generators and in instrumentation systems, such as spectrum analyzers and modulation analyzers. A basic configuration of a frequency synthesizer is shown in Figure 1-3 [10]. Besides a PLL, it also includes a very stable crystal oscillator with a divide by *N*-programmable divider in the feedback loop. The programmable divider divides the output of the VCO by *N* and locks to the reference frequency generated by a crystal oscillator. This is a traditional frequency synthesizer, also known as an Integer-*N* frequency synthesizer.

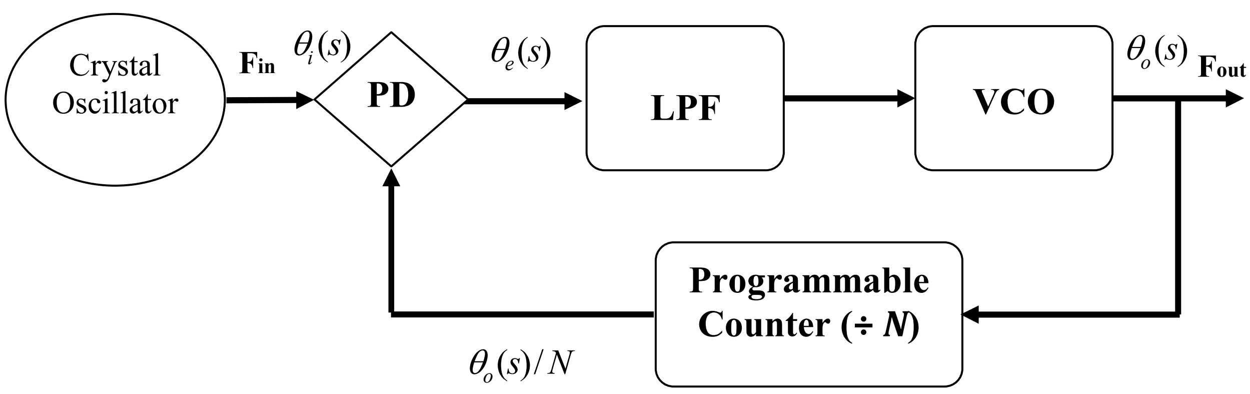

Figure 1-3: basic Frequency Synthesizer

The output frequency of VCO is a function of the control voltage of the filter. The Output of the phase comparator, which is proportional to the phase difference between the signals applied at its two inputs, control the frequency of the VCO. So the phase comparator input from the VCO through the programmable divider remains in phase with the reference input of crystal oscillator. The VCO frequency is thus maintained at NF_r. This relation can be expressed as

$$F_r = \frac{F_o}{N} \tag{1.1}$$

This implies that the output frequency is equal to

$$F_o = NF_r \tag{1.2}$$

Using this technique one can produce a number of frequencies separated by F_r and a multiple of *N*. For example if the input frequency is 34 KHz and the *N* is selected to be 28 (a single integer) then the output frequency will be 0.952 MHz. In the same way, if *N* is a range of numbers, the output frequencies will be in the proportional range. This basic technique can be used to develop a frequency synthesizer from a single reference frequency. This is the most basic form of a frequency synthesizer using phase locked loop technique.

From the Figure 1-3, note that θ_i is the phase input, θ_e the phase error, and θ_o output phase. Phase error (phase detector output) can be calculated from

$$\theta_e(s) = \frac{1}{1 + G(s)H(s)} \theta_i(s) \tag{1.3}$$

and the VCO output can be calculated from

$$\theta_o(s) = \frac{G(s)}{1 + G(s)H(s)} \theta_i(s), \tag{1.4}$$

where $G(s)$ is the product of the individual feed forward transfer functions, and $H(s)$ is the product of the individual feedback transfer functions.

One of the drawbacks of a traditional frequency synthesizer [23], also known as an integer-*N* frequency synthesizer, is that the output frequency is constrained to be *N* times the reference frequency. If the output frequency is to be adjusted by changing *N*, which is constrained by the divider to be an integer, then the output frequency resolution is equal to the reference frequency. If a fine frequency resolution is desired, then the reference frequency must be small. This in turn limits the loop bandwidth as set by the loop filter, which must be at least 10 times smaller than the reference frequency to prevent signal components at the reference frequency from reaching the input of the VCO and modulating the output frequency, creating spurs or sidebands at an offset equal to the reference frequency and its harmonics. A low loop bandwidth is undesirable because it limits the response time of the synthesizer to changes in *N*. In addition, the loop acts to suppress the phase noise in the VCO at offset frequencies within its bandwidth, so reducing the loop bandwidth acts to increase the total phase noise at the output of the VCO. The constraint on the loop bandwidth imposed by the required frequency resolution is eliminated if the divide ratio *N* is not limited to be an integer. This is the idea behind fractional-*N* synthesis. In the modern communication systems such as WiMAX system the fractional-*N* synthesizer was used.

Linear Matrix Inequalities (LMIs) is control method which widely used in control systems to solve many control problems. I used this method to design the loop filter in frequency synthesizer. The next section presents the overview about the LMI method and semidefinite programming (SDP).

1.4 Linear Matrix Inequalities (LMIs)

A wide variety of problems arising in control systems can be reduced to a few standard convex or quasi-convex optimization problems involving linear matrix inequalities (LMIs)**,** that is constraints of the form

$$F(x) \stackrel{\Delta}{=} F_0 + \sum_{i=1}^{m} x_i F_i > 0, \qquad (1.5)$$

where $x \in R^m$ is the variable and $F_i = F_i^T \in R^{n\times n}$, $i = 0,...,m$, are given.

Although the LMI form appears very specialized, it is widely encountered in control systems. List of many comprehensive examples is found in Boyd et al [7]. Since these resulting optimization problems can be solved numerically very efficiently, there are in special cases few analytical solutions to LMI optimization problems.

Indeed, the recent popularity of LMI optimization for control can be directly traced to the recent breakthroughs in interior point methods for LMI optimization. The growing popularity of LMI methods for control is also evidenced by the large number of publications in recent control conferences [7].

Semi-definite Programming (SDP)

Semidefinite programming (SDP) is a branch of convex programming (CP) that has been a subject of intensive research since the early 1990's [8]. The continued interest in SDP has been motivated mainly by two reasons.

First, many important classes of optimization problems such as linear-programming (LP) and convex quadratic-programming (QP) problems can be viewed as SDP problems, and many CP problems of practical usefulness that are neither LP nor QP problems can also be formulated as SDP problems. Second, several interiorpoint methods that have proven efficient for LP and convex QP problems have been extended to SDP in recent years. The semidefinite programming problem (SDP) is essentially an ordinary linear program where the non negativity constraint is replaced by a semidefinite constraint on matrix variables [9].

A semidefinite program is an optimization problem of the following form [9]:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & F(x) \geq 0, \end{array} \tag{1.6}$$

where

$$F(x) \overset{\Delta}{=} F_0 + \sum_{i=1}^{m} x_i F_i \tag{1.7}$$

The problem data are the vector $c \in R^m$ and m + 1 symmetric matrices $F_0, \cdots, F_m \in R^{n\times n}$. The inequality sign in $F(x) \geq 0$ means that $F(x)$ is a positive semidefinite, i.e., $z^T F(x) z \geq 0$ for all $z \in R^n$. We call the inequality $F(x) \geq 0$ a linear matrix inequality and the equation (1.6) a semidefinite program.

A semidefinite program is a convex optimization problem since its objective and constraint are convex: if $F(x) \geq 0$ and $F(y) \geq 0$, then, for all $\lambda, 0 \leq \lambda \leq 1$, $F(\lambda x + (1-\lambda y)) = \lambda F(x) + (1-\lambda) F(y) \geq 0$. Figure 1-4 depicts a simple example with $x \in R^2$ and $F_i \in R^{7\times 7}$ "In [9]". Our goal here is to give the reader a generic picture that shows some of the features of semidefinite programs, so the specific values of the data are not relevant. The boundary of the feasible region is shown as the solid curve.

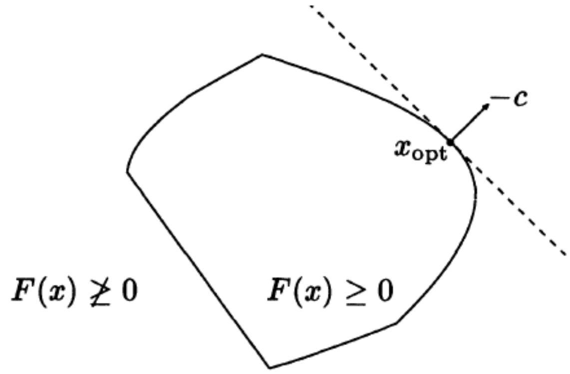

Figure 1-4: A simple semidefinite program with $x \in R^2$ and $F_i \in R^{7\times 7}$ [9]

The feasible region, i.e., $\{x \mid F(x) \geq 0\}$ consists of this boundary curve along with the region it encloses. Very roughly speaking, the semidefinite programming problem is to move as far as possible in the direction -*c*, while staying in the feasible region. For this semidefinite program there is one optimal point, X_{opt} "see [9]".

This simple example demonstrates several general features of semidefinite programs. We have already mentioned that the feasible set is convex. Note that the optimal solution X_{opt} is on the boundary of the feasible set, i.e., $F(x_{opt})$ is singular; in the general case there is always an optimal point on the boundary (provided the problem is feasible). In this example, the boundary of the feasible set is not smooth. It is piecewise smooth: it consists of two line segments and two smooth curved segments. In the general case the boundary consists of piecewise algebraic surfaces. Skipping some technicalities, the idea is as follows. At a point where the boundary is smooth, it is defined locally by some specific minors of the matrix $F(x)$ vanishing. Thus the boundary is locally the zero set of some polynomials in $x_1, \cdots, x_m$, i.e., an algebraic surface.

In the next section we discussed the problem which we need to solve it.

1.5 The Problem

The objective from this thesis is to design PLL loop filter that accomplish the requirements to work properly and efficiently with Fixed WiMAX system. The design is based on optimization particularly Linear Matrix Inequalities (LMI) technique. Much of the research effort in the application of LMI optimization has been directed towards problems from control theory while, many of the underlying techniques extend to problems from other areas of engineering as well, for instant, wireless applications. The problem to design and optimize the loop filter using LMI is new in that it is used for Fixed WiMAX system.

Fixed WiMAX systems use digital low pass filter (IIR or FIR) as a main component of fractional-*N* synthesizer. In other words, the problem is to design IIR filter or FIR filter to replace the loop filter in the PLL block.

The designed loop filter has to be stable and compatible with:

1. Frequency range used for Fixed WiMAX systems (3.5 - 5.8) GHz.
2. Line-of-site requirement.
3. Small settling time.
4. Small overshoot.
5. Small raising time.

In this thesis, we will utilize the semidefinite programming (SDP) to design stable minimax IIR & FIR digital filters. In wireless systems such as Fixed WiMAX and Mobile WiMAX, we do not use simple PLL architecture, instead we can use integer-*N* frequency synthesizer and for more modern systems we use fractional-*N* frequency synthesizer.

In our design, the VCO noise is neglected. fractional-*N* frequency synthesizer is used instead of integer-*N* frequency synthesizer to reduce noise resulted from the factor *N*. Traditional frequency synthesizers use low-pass analog filter to eliminate the high frequency components.

In the new design, IIR or FIR digital low-pass filter is used to eliminate the high frequency components resulting in more immunity to noise. Thesis Organization is presented in the next section.

1.6 Thesis Organization

In this thesis, we introduced six chapters. Chapter 2 covers literature review of PLL loop filter designs and optimizations. Theoretical background on Fixed WiMax standard is presented and outlined in chapter 3. The design of fractional-*N* frequency synthesizer using IIR and FIR digital filters is discussed in chapter 4. Chapter 5 displays the solution of the problem using the toolbox, *SeDuMi*, stands for self-dual minimization with comparison with other techniques. Finally the conclusion and suggestions for future work is given in chapter 6.

CHAPTER 2

Literature Review

Phase locked loops (PLLs) have been around for many years. In many respects, the field is mature, with widespread application to almost every type of communication and storage device. An incomplete list of specific tasks accomplished by PLLs include carrier recovery, clock recovery, tracking filters, frequency and phase demodulation, phase modulation, frequency synthesis, and clock synchronization. PLLs find themselves into a huge set of applications, from radio and television, to virtually every type of communications (wireless, telecom, data communication), to virtually all types of storage device, to noise cancellers, and other applications. Because the widespread use by the public of such devices, control engineers interested to researching in the PLL field and trying to have a good results. In this thesis I present the new method to design loop filter in the PLL.

Analysed and designed analog PLLs loop filters were introduced by many techniques in [11], [12], [14], [15], [16] and [17]. The main objective for those techniques is reducing time delay and noise level. One of the main drawbacks of those techniques is designing complexity. Colleran, et al. described the global optimization of PLL circuits using geometric programming (GP) [18]. Using this technique, the PLL design cycle was reduced from a matter of weeks to a matter of hours.

In the 1970's [5], the applications of PLL were widely used in modem communication systems. Fahim and Elmasry presented a fast lock digital PLL frequency synthesizer for wireless applications [19]. Ghartemani et al. [20], introduced an alternative structure for a PLL system. This structure is called

Quadrature PLL (QPLL). The QPLL is advantageous for communication system applications which with quadrature modulation techniques. Mao et al. [21], presented a new method for improving the phase tracking performance of GPS receivers. They are used fuzzy logic controller (FLC) to design digital phase-locked loops DPLL.

One of The main PLL applications in modern communication system is used as fractional-*N* frequency synthesizers. In this thesis I use the fractional-*N* frequency synthesizer as the PLL application which used in fixed WiMAX communication system. Many methodologies for designing fractional-*N* frequency synthesizers are presented in [24], [25], and [26]. The fractional-*N* frequency synthesizers were designed in general form.

The loop filter is the main components in fractional-*N* frequency synthesizer. They are many different methods to design the loop filters in fractional-*N* frequency synthesizer. One of the main methods which used to design those filters is the linear matrix inequality (LMI) method as in [22] and [28]. Those designed filters were used in fractional-*N* frequency synthesizer in transceiver device for GPS communication systems.

A wide variety of problems arising in control systems can be reduced to a handful of standard convex and quasi-convex optimization problems that involve matrix inequalities. LMI problems have many forms such as SDP problem which solved by many algorithms. Abbas-Turki, et al. [13], presented an LMI formulation for designing controllers according to time response and stability margin constants which translated using convex programming. They used the Cutting Plane Algorithm (CPA) to solve their control problem. The nonlinear PLL design using SDP was illustrated by Wang, et al. [27]. This design approached was based on the polynomial nonlinear model of the PLL system.

Filter design is one of the most important problems in Signal Processing. Many techniques are used to design digital filters such as LMI method. Al-Baroudi in 1997 [29] formulated the FIR filter design problem as a Linear Objective Optimization problem with LMI constraints, and solved it. He also formulated the IIR filter design problem as a Linear Objective Optimization problem with LMI constraint and with iterative scheme to solve it. Another contribution was the introduction of the frequency selection algorithm that reduced the number of LMI's solved to reach the optimal solution. He also introduced the formulation of the FIR/IIR optimal power spectrum approximation problem as linear objective optimization problem with LMI constraint, and solved it. Finally he introduced an LMI-based model reduction technique.

FIR filter design via SDP and spectral factorization was presented by Wu, et al. [30]. They presented a new SDP approach to FIR filter design with arbitrary upper and lower bounds on the frequency response magnitude. It was shown that the constraints can be expressed as LMIs. Using this LMI formulation, we can cast several interesting filter design problems as convex or quasi-convex optimization problems. Many other extensions that were not discussed in that paper can be handled in the same framework, such as, maximum stopband attenuation or minimum transition-band width FIR design given magnitude bounds, or even linear array beam-forming.

Linear programming design of IIR digital filters with arbitrary magnitude function was presented by Rabiner, et al. [31]. Their paper discussed the use of linear programming techniques for the design of IIR digital filters. In particular, it was shown that, in theory, a weighted equiripple approximation to an arbitrary magnitude function can be obtained in a predictable number of applications of the simplex algorithm of linear programming. When one implements the design algorithm, certain practical difficulties (e.g., coefficient sensitivity) limit the range of filters which can be designed using this technique.

Al-Quqa achieved optimal design of PLL in Mobile WiMAX system [32]. A new loop filter design method for PLLs was introduced taking into consideration various design objectives. Digital filters, FIR were designed using linear programming and convex programming (SDP), but IIR was designed using linear programming only. Al-Quqa recommended the digital FIR low pass filter for mobile WiMax systems because it improves the transient behavior.

Antoniou [33] explained that the linear-phase FIR filters are often designed very efficiently using the weighted-Chebyshev method which is essentially a minimax method based on the Remez exchange algorithm. The disadvantage of the proposed method that certain types of filters cannot be designed with this method such as low-delay FIR filters with approximately constant passband group delay.

Those filters can be designed using a minimax method based on SDP, as illustrated by Antoniou and Lu. [8]. They used the SDP approach to design FIR filters with arbitrary amplitude and phase responses. They solved SDP problem by primal-dual path-following algorithm. They proved the SDP minimax approach actually obtained the unique best equiripple linear-phase design.

Lu described an SDP-based method for the design of stable minimax IIR filter [34]. The stability of the filter was assured by using a single LMI constraint derived from the well-known Lyapunov theory. He was used the primal-dual path-following algorithm to solved the SDP problem. He found the presented design offers improved performance as well as a reduced group delay. In addition, the presented filter lead to reduced computational complexity in the implementation of the filter.

Previous studies mentioned above have made important contributions to this challenging problem; however, none of the published studies appear to have completely accounted of my thesis subject which depend on the designing the FIR and IIR digital filters by reformulated the control problem as SDP problem with LMI constraints and solved the SDP problem by SeDuMi toolbox software and used those designed filters as loop filters in the fractional-*N* frequency synthesizers for Fixed WiMAX application.

CHAPTER 3

Fixed WiMAX

The growing demand for broadband services on a global scale is incontestable. Businesses, public institutions and private users regard it as an enabling technology and it has become a requirement for delivering communication services in the information age. The desire for bandwidth-intensive Internet access and other voice and data services has never been greater across all geographies and market segments, by using Broadband Wireless Access (BWA) technology.

Broadband wireless is a technology that promises high-speed (minimally, several hundred kilo bits per second) data transmissions occurring within an infrastructure of more or less fixed points, including both stationary subscriber terminals and service provider base stations, which themselves constitute the hubs of the network [35].

There are two fundamentally different types of broadband wireless services. The first type attempts to provide a set of services similar to that of the traditional fixed-line broadband but using wireless as the medium of transmission. This type, called fixed wireless broadband, can be thought of as a competitive alternative to DSL or cable modem. The second type of broadband wireless, called mobile broadband, offers the additional functionality of portability, nomadicity, and mobility. WiMAX (worldwide interoperability for microwave access) technology is designed to accommodate both fixed and mobile broadband applications [35].

WiMAX is a scaleable wireless platform for constructing alternative and complementary broadband networks. WiMAX systems are expected to deliver broadband access services to residential and enterprise customers in an economical way [35].

WiMAX is a certification that denotes interoperability of equipment built to the IEEE 802.16 or compatible standard [36]. The IEEE 802.16 working group develops standards that address two types of used models: a fixed usage model (IEEE 802.16-2004) and a portable (mobile) usage model (IEEE 802.16 e-2005).

3.1 Evolution of WiMAX

WiMAX technology has been developed in four stages [36]:

1. Narrowband wireless local-loop systems.
2. First-generation line-of-sight (LOS) broadband systems.
3. Second-generation non-line-of-sight (NLOS) broadband systems.
4. Standards-based broadband wireless systems.

3.1.1 Narrowband Wireless Local-Loop Systems

Naturally, the first application for which a wireless alternative was developed and deployed was voice telephony. These systems, called wireless local-loop (WLL), were quite successful in developing countries whose high demand for basic telephone services could not be served using existing infrastructure [36].

During 2001, several small start-up companies focused solely on wireless Internet- access services deployed systems in the license-exempt 900 MHz and 2.4 GHz bands [36].

Most of these systems required antennas to be installed at the customer premises, either on rooftops or under the eaves of their buildings. Deployments were limited mostly to select neighborhoods and small towns. These early systems typically offered speeds up to a few hundred kilo bits per second. Later evolutions of license-exempt systems were able to provide higher speeds [36].

3.1.2 First-Generation Line-Of-Sight (LOS) Broadband Systems

In the late 1990s, one of the more important deployments of wireless broadband happened in the so-called multichannel multipoint distribution services (MMDS) band at 2.5 GHz [36]. The MMDS band was historically used to provide wireless cable broadcast video services, especially in rural areas where cable TV services were not available. The advent of satellite TV ruined the wireless cable business, and operators were looking for alternative ways to use this spectrum. The first generation of these fixed broadband wireless solutions was deployed using the same towers that served wireless cable subscribers. These towers were typically several hundred feet tall and enabled LOS coverage to distances up to 35 miles, using high-power transmitters. First-generation MMDS systems required that subscribers install at their premises outdoor antennas high enough and pointed toward the tower for a clear LOS transmission path [36].

3.1.3 Second-Generation Non-Line-Of-Sight (NLOS) Broadband Systems

Second-generation broadband wireless systems were able to overcome the LOS issue and to provide more capacity. This was done through the use of a cellular architecture and implementation of advanced-signal processing techniques to improve the link and system performance under multipath conditions [36].

Most of new systems could perform well under non-line-of-sight conditions, with customer-premise antennas typically mounted under the eaves or lower. The NLOS problem were solved by using such techniques as orthogonal frequency division multiplexing (OFDM), code division multiple access (CDMA), and multiantenna processing. Some systems demonstrated satisfactory link performance over a few miles to desktop subscriber terminals without the need for an antenna mounted outside. A few megabits per second throughput over cell ranges of a few miles had become possible with second generation fixed wireless broadband systems.

3.1.4 Standards-Based Broadband Wireless Systems

In 1998, the Institute of Electrical and Electronics Engineers (IEEE) formed a group called 802.16 to develop a standard for what was called a wireless metropolitan area network, or wireless MAN [36]. Originally, this group focused on developing solutions in the 10 GHz to 66 GHz band, with the primary application being delivering high-speed connections to businesses that could not obtain fiber. The IEEE 802.16 group produced Wireless MAN-SC standard that was approved in December 2001.

After completing this standard, the group started work on extending and modifying it to work in both licensed and license-exempt frequencies in the 2 GHz to 11 GHz range, which would enable NLOS deployments.

This amendment, IEEE 802.16a, was completed in 2003; further revisions to 802.16a were made and completed in 2004. This revised standard, IEEE 802.16-2004, replaces 802.16, 802.16a, and 802.16c with a single standard, which has also been adopted as the basis for HIPERMAN (high-performance metropolitan area network) by ETSI (European Telecommunications Standards Institute). In 2003, the 802.16 group began work on enhancements to the specifications to allow vehicular mobility applications. That revision, 802.16e, was completed in December 2005 and was published formally as IEEE 802.16e-2005.

There are many advantages of systems based on 802.16, e.g. the ability to provide service even in areas that are difficult for wired infrastructure to reach and the ability to overcome the physical limitations of traditional wired infrastructure. The standard will offer wireless connectivity of up to 30 miles. The major capabilities of the standard are its widespread reach, which can be used to set up a metropolitan area network, and its data capacity of 75 Mbps [35].

This high-speed wireless broadband technology promises to open new, economically viable market opportunities for operators, wireless Internet service providers and equipment manufacturers.

The flexibility of wireless technology, combined with high throughput, scalability and long-range features of the IEEE 802.16 standard helps to fill the broadband coverage gaps and reach millions of new residential and business customers worldwide (Figure 3.1) [35].

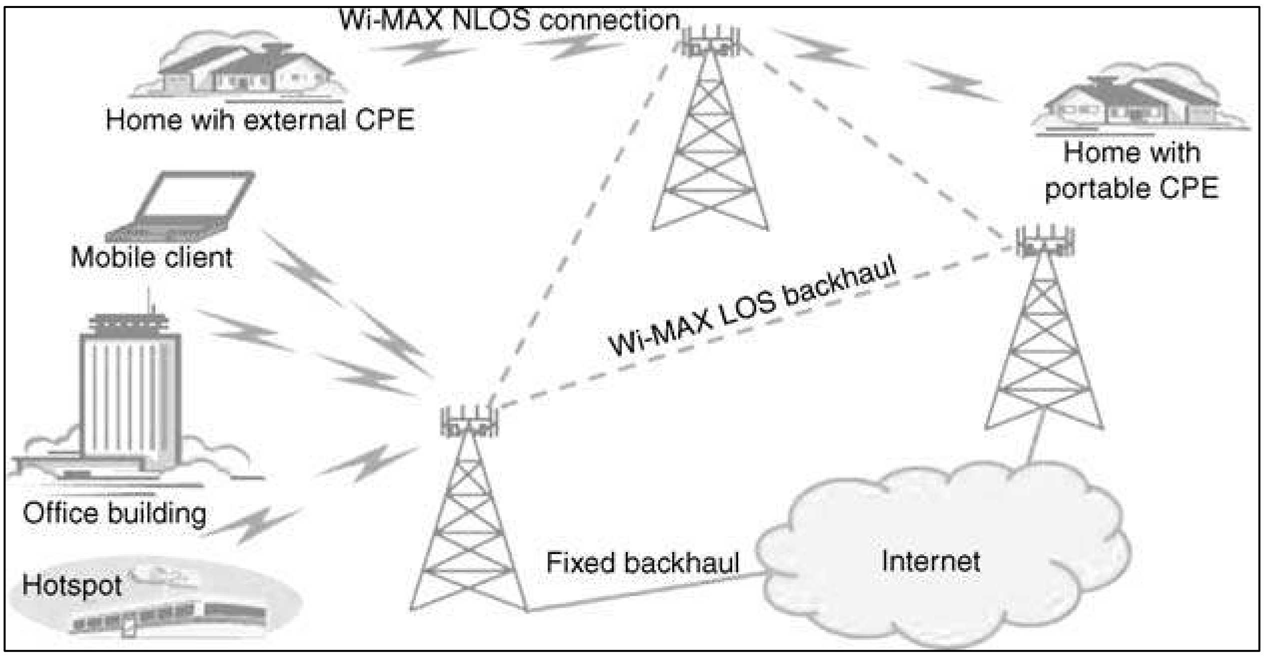

Figure 3.1: WiMAX solutions [4]

3.2 Fixed WiMAX vs. Mobile WiMAX

We can summarize the differences between two WiMAX systems as follow [38]:

3.2.1 Mobility Management and Hand Offs

The simplest explanation for the difference between the fixed and mobile variants of WiMAX boil down to the fact that the mobile variant enables a hand-off from one base station to another as the user, in one session, moves from the coverage zone of one base station to another. This is also known as "mobility management".

3.2.2 OFDM vs. SOFDMA

Fixed WiMAX uses OFDM. Orthogonal Frequency Division Multiplexing (OFDM) breaks the wireless carrier into 256 sub-carriers (or little waves). The mobile variant of WiMAX uses OFDMA. Orthogonal Frequency Division Multiple Access (OFDMA) breaks the carrier into even more sub carriers (up to 2048 sub carriers). The advantage of this is better propagation and potentially improved building penetration (although other factors such as frequency and power come into play here as well) relative to OFDM. The use of OFDMA should also enable the use of smaller, less costly subscriber devices including PC cards and USB devices.

3.2.3 Costs

Mobile WiMAX base stations more expensive than fixed WiMAX base stations [38]. The service provider must weigh cost benefit of the more expensive base station relative to their markets.

Table 3-1 Fixed WiMAX vs. Mobile WiMAX

	Fixed WiMAX	Mobile WiMAX
Frequency(GHz)	3.5, 5.8	2.3, 2.5, 3.5, etc
Channel (MHz)	3.5,7,10,14	3.5,7,8.75,10,14, etc
Duplexing	TDD/FDD	TDD/FDD
Multiplexing	OFDM	OFDMA

3.2.4 Quality of Service

The fixed variant of WiMAX provides four categories for prioritizing traffic. This means that time sensitive traffic such as voice and video get priority over non-time sensitive traffic such as data. Table 3-1 shows a comparison between fixed and mobile WiMax technologies [39].

3.3 WiMAX Components

Typically, a WiMAX system consists of two parts: a WiMAX base station and a WiMAX receiver, also referred as customer premise equipment (CPE). The backhaul connects the system to the core network as shown in Figure 3.2 [35].

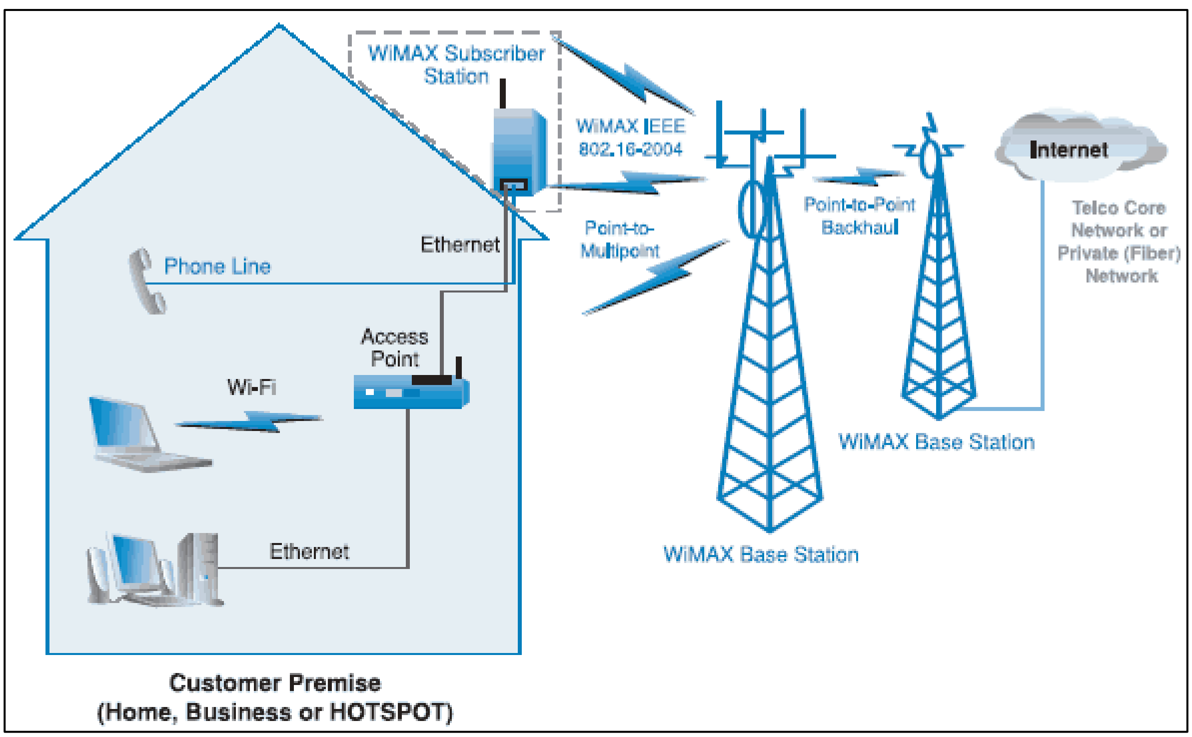

Figure 3.2: WiMAX technology [4]

3.3.1 The WiMAX base station

A WiMAX base station consists of indoor electronics and a WiMAX tower. The WiMAX base stations would use the MAC layer defined in the standard – a common interface that makes the networks interoperable – and would allocate uplink and downlink bandwidth to subscribers according to their needs, on an essentially real-time basis. Each base station provides wireless coverage over an area called a cell. The maximum radius of a cell is theoretically 50 km (depending on the frequency band chosen); however, typical deployments will use cells of radii from 3 to 10 km. As with conventional cellular mobile networks, the base-station antennas can be omni directional, giving a circular cell shape, or directional to give a range of linear or sectoral shapes for point-to-point use or for increasing the network's capacity by effectively dividing large cells into several smaller sectoral areas (Figure 3.3) [35].

Figure 3.3: WiMAX base station

3.3.2 The WiMAX receiver

A WiMAX receiver may have a separate antenna (i.e. receiver electronics and antenna are separate modules) or could be a stand-alone box or a PCMCIA card that sits in your laptop or computer. Access to a WiMAX base station is similar to accessing a wireless access point in a WiFi network, but the coverage is greater. So far one of the biggest deterrents to the widespread acceptance of BWA has been the cost of CPE. This is not only the cost of the CPE itself, but also the installation cost. Historically, proprietary BWA systems have been predominantly line-of-site, requiring highly skilled labour to install and 'turn up' a customer device. The concept of a self-installed CPE has been the Holy Grail for BWA from the beginning. With the advent of WiMAX this issue seems to be resolving (Figure 3.4) [35].

Figure 3.4: WiMAX receivers

3.3.3 Backhaul

Backhaul refers both to the connection from the access point back to the provider and to the connection from the provider to the core network. A backhaul can deploy any technology and media provided. It connects the system to the backbone. In most of the WiMAX deployment scenarios, it is also possible to connect several base stations to one another using high-speed backhaul microwave links.

This would also allow for roaming by a WiMAX subscriber from one base station coverage area to another, similar to the roaming enabled by cell phones (Figure 3.2) [35].

3.4 WiMAX Working

The backhaul of the WiMAX is based on the typical connection to the public wireless networks by using optical fibre, microwave link, cable or any other high speed connectivity. In few cases such as mesh networks, Point-to-Multi-Point (PMP) connectivity is also used as a backhaul. Ideally, WiMAX should use Point-to-Point antennas as a backhaul to join subscriber sites to each other and to base stations across long distance. A base station serves subscriber stations using Non-Line-of-Sight (NLOS) or LOS Point-to-Multi-Point connectivity; and this connection is referred to as the last mile communication. Ideally, WiMAX should use NLOS Point-to-Multi-Point antennas to connect residential or business subscribers to the Base Station (BS). A subscriber station typically serves a building using wired or wireless LAN

3.5 Fixed WiMAX applications

Applications using a fixed wireless solution can be classified as point-to-point or point-to-multipoint. Point-to-point applications include interbuilding connectivity within a campus and microwave backhaul. Point-to-multipoint applications include:

1. Broadband for residential, small office/home office (SOHO), and small- to medium-enterprise (SME) markets.
2. T1 or fractional T1-like services to businesses.
3. Wireless backhaul for Wi-Fi hotspots.

Figure 3.5 illustrates the various point -to- multipoint applications.

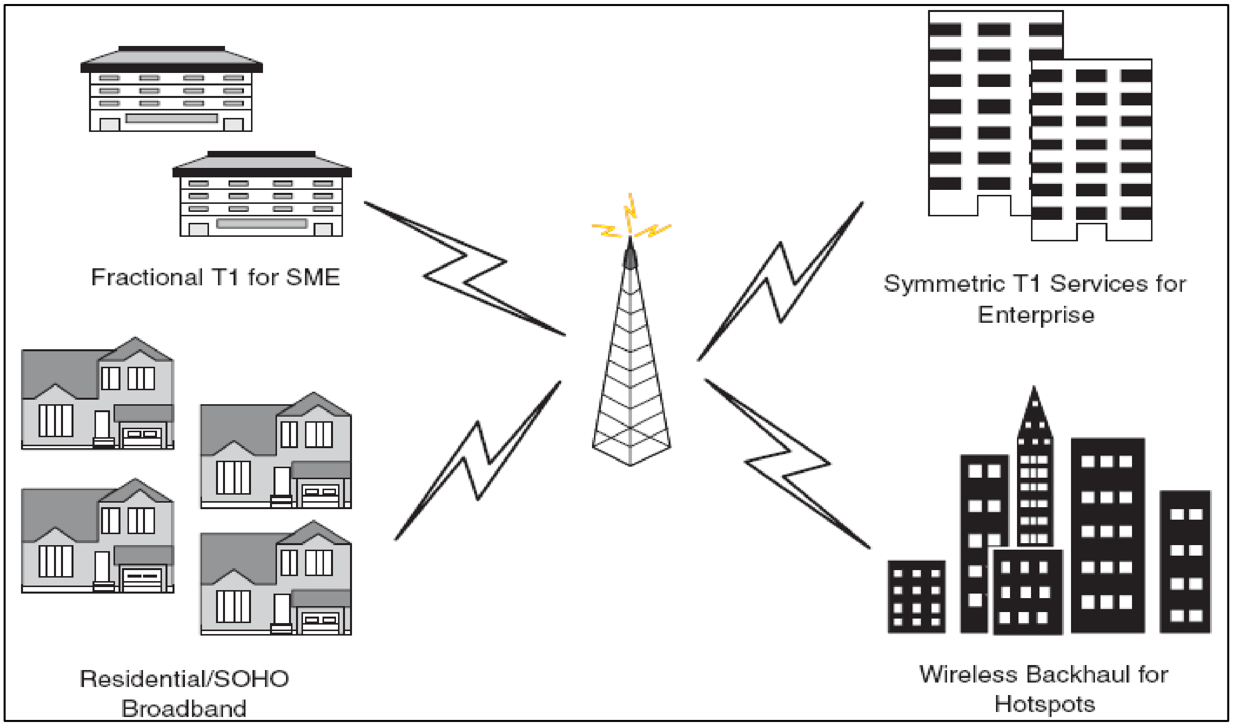

Figure 3.5: Point-to-multipoint WiMAX applications [1]

3.5.1 Consumer and small-business broadband

Clearly, one of the largest applications of WiMAX in the near future is likely to be broadband access for residential, SOHO, and SME markets. Broadband services provided using fixed WiMAX could include high-speed Internet access, telephony services using voice over IP, and a host of other Internet-based applications. Fixed wireless offers several advantages over traditional wired solutions. These advantages include lower entry and deployment costs; faster and easier deployment and revenue realization; ability to build out the network as needed; lower operational costs for network maintenance, management, and operation; and independence from the incumbent carriers [35].

3.5.2 T1 emulation for business

The other major opportunity for fixed WiMAX in developed markets is as a solution for competitive T1/E1, fractional T1/E1, or higher-speed services for the business market. Given that only a small fraction of commercial buildings worldwide have access to fiber, there is a clear need for alternative high-bandwidth solutions for enterprise customers. In the business market, there is demand for symmetrical T1/E1 services that cable and DSL have so far not met the technical requirements for Traditional telco services continue to serve this demand with relatively little competition. Fixed broadband solutions using WiMAX could potentially compete in this market and trump landline solutions in terms of time to market, pricing, and dynamic provisioning of bandwidth [35].

3.5.3 Backhaul for Wi-Fi hotspots

Another interesting opportunity for WiMAX in the developed world is the potential to serve as the backhaul connection to the burgeoning Wi-Fi hotspots market. In the United States and other developed markets, a growing number of Wi-Fi hotspots are being deployed in public areas such as convention centers, hotels, airports, and coffee shops [36].

The Wi-Fi hotspot deployments are expected to continue to grow in the coming years. Most Wi- Fi hotspot operators currently use wired broadband connections to connect the hotspots back to a network point of presence. WiMAX could serve as a faster and cheaper alternative to wired backhaul for these hotspots. Using the point-to-multipoint transmission capabilities of WiMAX to serve as backhaul links to hotspots could substantially improve the business case for Wi-Fi hotspots and provide further momentum for hotspot deployment. Similarly, WiMAX could serve as 3G (third-generation) cellular backhaul [35].

A potentially larger market for fixed broadband WiMAX exists outside the United States, particularly in urban and suburban locales in developing economies—China, India, Russia, Indonesia, Brazil and several other countries in Latin America, Eastern Europe, Asia, and Africa—that lack an installed base of wireline broadband networks. National governments that are eager to quickly catch up with developed countries without massive, expensive, and slow network rollouts could use WiMAX to leapfrog ahead. A number of these countries have seen sizable deployments of legacy WLL systems for voice and narrowband data. Vendors and carriers of these networks will find it easy to promote the value of WiMAX to support broadband data and voice in a fixed environment [35].

CHAPTER 4

Design of PLL Filter

Phase-locked loops are widely used in radio, telecommunications, computers and other electronic applications. They are used in WiMAX as a frequency synthesizer. Frequency synthesizer defined as a system that generates one or many frequencies derived from a single time base (frequency reference) of a crystal oscillator, in such a way that the ratio of the output to the reference frequency is a rational fraction.

The Loop Filter, VCO, very stable crystal oscillator with a divide by *N*-programmable divider in the feedback loop, and phase detector (PD) together comprise a frequency synthesizer as shown in Figure 1-3.

The programmable divider divides the output of the VCO by *N* and locks to the reference frequency generated by a crystal oscillator. To gain some design experience, some more insight and to compare results, consider a real design problem of a frequency synthesizer loop filter with the general specifications as shown in Table 4-1 [32].

Table 4-1: Design Specifications

DESIGN SPECIFICATIONS	
PARAMETER	**SPECIFICATION**
Frequency Range	3.5 GHz – 5.8 GHz
Resolution	240 KHz
Overshoot Less than	2%
Settling time Less	Less than 5µs

The design process is divided into several stages. We first present the overall block of frequency synthesizer, then select the integer value of *N* according to reference frequency and resolution [32].

The next step is the design of digital IIR low pass loop filter. IIR low pass loop filter design is carried out using Semi-Definite programming (SDP) utilizing Linear Matrix Inequalities formulation (LMI). After that, FIR low-pass loop filter will be designed. FIR low pass loop filter design is carried out using SDP utilizing LMI formulation. In the next chapter, we will implement the algorithm and simulate the designed filters using our generated MATLAB codes.

4.1 Fractional-N PLL block diagram

The fractional-N PLL block diagram is showed in Figure 4-1,

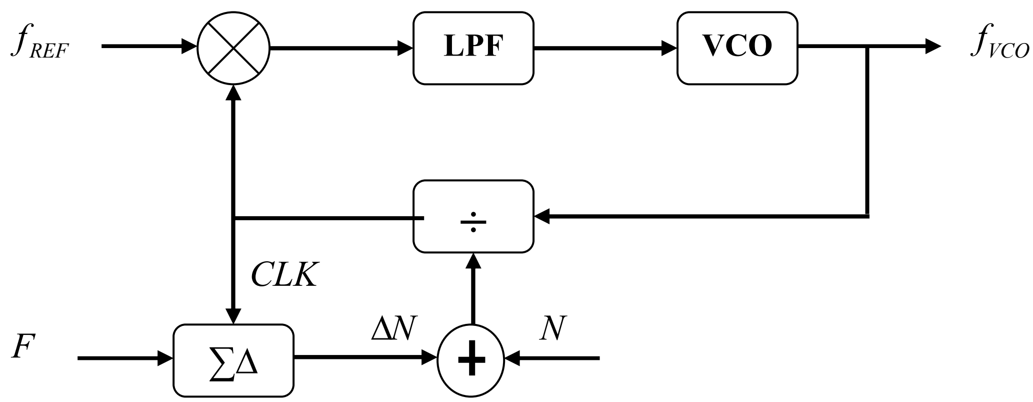

Figure 4-1: Basic Fractional-N PLL Block Diagram

The fractional-*N* PLL consists of:

1. Phase/Frequency detector which is assumed to be XOR type.
2. Loop filter which is the objective of our design, is a low-pass filter (LPF).
3. Voltage Control Oscillator (VCO).

We utilized the same procedures which has been used in [32], but for Fixed WiMAX frequencies (3.5-5.8 GHz). We first begin the design with integer-*N* PLL: 240 kHz Reference pushes *N* from 14583 to 24167 (3500 / 0.240) to (5800 /0.240).

As a result the loop filter cutoff (less than 12.5 KHz) produces long settling time and VCO phase noise increased by $20*\log_{10}(N) \approx 89dB$.

To overcome the previous drawback, we use the Multi-Modulus Fractional PLL with these properties [32]:

• Fractional value between *N* and *2N-1* (64-127).

• Sigma Delta Modulator (programmable resolution).

• Large reference (20 MHz) for good tradeoff with settling time.

• Reduced *N* impact on phase noise by 45 dB over integer *N*.

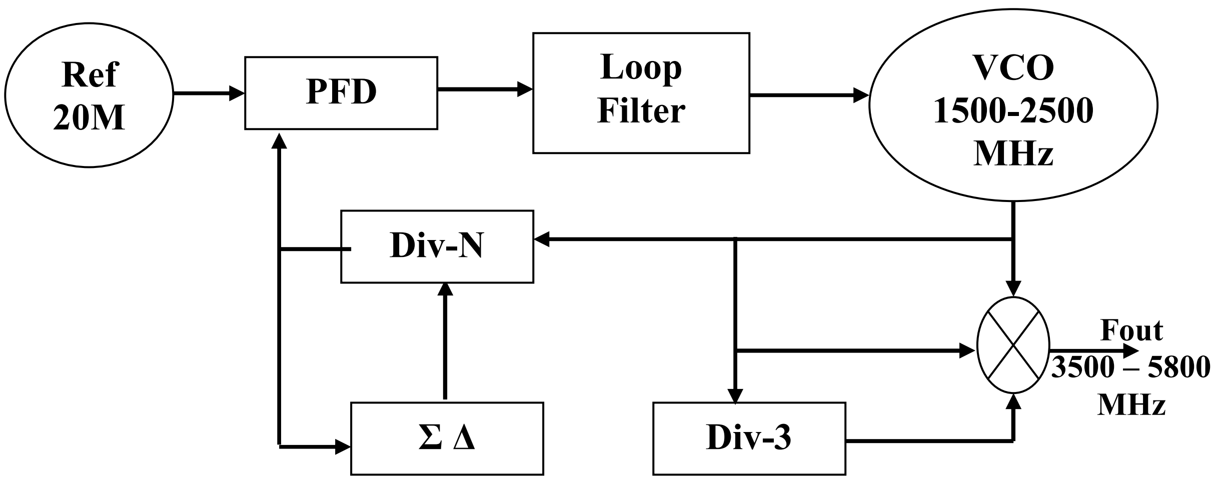

Figure 4-2: Designed Fractional-*N* Synthesizer Block

Example 4.1:

To produce 3500 MHz, we produce 1533 MHz (from VCO) and then upconvert it to 3500 MHz (1533 MHz * 2.3 ≈ 3500 MHz). The 1533 MHz can be produced with *N* = 76 and a fraction = 0.65 (means that 20 MHz * (76 + 0.65) = 1533 MHz).

As a result, for *N* = (76 ~ 125), it can produce frequency range (1533MHz ~ 2533MHz), which can be upconverted to (3500MHz ~ 5800MHz).

Figure 4-2 shows the fractional-*N* PLL design block diagram, with *N* range from 76 to 125 and $\sum\Delta$ is the fraction. From the previous analysis and compared with [32], we can conclude the differences between the designed fractional-*N* synthesizer in Mobile WiMAX and Fixed WiMAX as following points:

- We used the same reference frequency (20 MHz) to designed fractional-*N* synthesizer in Mobile WiMAX and Fixed WiMAX.
- The value of (Div-*N*) in Mobile WiMAX is (76-90) and in Fixed WiMAX is (76-125).
- In Fixed WiMAX, we used Div-3 instead Div-2 in Mobile WiMAX.
- We used in Fixed WiMAX VCO with oscillation frequency range (1500-2500 MHz), but in Mobile WiMAX the range is (1533 – 1800 MHz).

Our goal now is to design the low pass loop filter in order to meet the previously mentioned requirements.

4.2 IIR Low-Pass Filter Design by LMI [34]

Recursive Infinite – Impulse – Response (IIR) digital filters offer improved selectivity, computation efficiency and reduce system delay compared to what can be achieved by nonrecursive Finite – Impulse – Response (FIR) digital filters of comparable approximation accuracy. The major drawbacks of IIR filter designs are that the linear phase response can be achieved only approximately and the design must handle the stability problem which dose not exist in the FIR case [34].

In this thesis, we used an SDP based method for the design of stable minimax IIR filter. In this method, the stability is ensured by imposing a single linear matrix inequality (LMI) constraint derived from the well-known Lyapunov theory [34].

The transfer function of the IIR filter to be designed is assumed to be of the form:

$$H(z) = \frac{A(z)}{B(z)}, \tag{4.1}$$

where,

$$A(z) = \sum_{i=0}^{N} a_i z^{-i}, \qquad (4.2)$$

and

$$B(z) = 1 + \sum_{i=1}^{K} b_i z^{-i}. \qquad (4.3)$$

4.2.1 LMI constraint for stability

The stability of a filter represented by transfer function $H(z)$ such as that in Eq. (4.1) is guaranteed if the zeros of polynomial $B(z)$ in Eq. (4.3) are strictly inside the unit circle. D is a $K \times K$ matrix in the controllable canonical form where the first row of D is the row vector $-b = -[b_1 \;\; b_2 ......b_K]$ and $b_1, b_2 ...., b_K$ are coefficients of *B(z)*. It can be shown that the zeros of $B(z)$ are the eigenvalues of the matrix D as below

$$D = \begin{bmatrix} -b_1 & -b_2 & \cdots & -b_{K-1} & -b_K \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \qquad (4.4)$$

Consequently, the filter is stable if the moduli of the eigenvalues are all strictly less than one. The well-known Lyapunov theory [34], states that D represents a stable filter if and only if there exists a positive definite matrix P such that $P - D^T PD$ is positive definite, i.e.,

$$F = \{P : P > 0 \; and \; P - D^T PD > 0\} \qquad (4.5)$$

is nonempty. Using simple linear algebraic manipulations, it can be verified that the set F in Eq. (4.5) can be characterized by

$$F = \left\{ P : \begin{bmatrix} P^{-1} & D \\ D^T & P \end{bmatrix} > 0 \right\} \quad (4.6)$$

Note that unlike the constraints in Eq. (4.5), matrix D in Eq. (4.6) (hence the coefficients of $B(z)$) appears affinely.

4.2.2 SDP formulation of the design problem [34]

Given a desired frequency response $H_d(\omega)$, a minimax design of a stable IIR filter can be obtained by finding a transfer function $H(z)$ such as Eq. (4.1) solves the constrained optimization problem

$$\text{minimize} \quad \max_{0 \le \omega \le \pi} \left[W(\omega) \left| H(\omega) - H_d(\omega) \right| \right], \quad (4.7)$$

$$\text{subject to:} \quad B(z) \neq 0 \quad for \left| z \right| \geq 1, \quad (4.8)$$

where $W(\omega)$ is a given weighting function and $H_d(\omega)$ can be written as

$$H_d(\omega) = H_r(\omega) + jH_i(\omega). \quad (4.9)$$

Where $H_r(\omega)$ and $H_i(\omega)$ denote the real and imaginary parts of $H_d(\omega)$, respectively.

The frequency response of the filter can be expressed as

$$H(\omega) = \frac{A(\omega)}{B(\omega)}, \quad (4.10)$$

where

$$A(\omega) = \sum_{n=0}^{N} a_n e^{-jn\omega} = a^T c(\omega) - ja^T s(\omega),$$

$$B(\omega) = 1 + \sum_{n=1}^{K} b_n e^{-jn\omega} = 1 + b^T \hat{c}(\omega) - jb^T \hat{s}(\omega),$$

$$a = [a_0 \;\; a_1 \ldots\ldots a_N]^T,$$
$$b = [b_1 \;\; b_2 \ldots\ldots b_K]^T,$$

$$c(\omega) = [1 \quad \cos\omega \cdots \cos N\omega]^T,$$
$$s(\omega) = [0 \quad \sin\omega \cdots \sin N\omega]^T,$$
$$\hat{c}(\omega) = [\cos\omega \;\; \cos 2\omega \cdots \cos K\omega]^T,$$
$$\hat{s}(\omega) = [\sin\omega \;\; \sin 2\omega \cdots \sin K\omega]^T.$$

If δ denotes the upper bound of the squared weighted error in Eq. (4.7), i.e.,

$$W^2(\omega)\left|H(\omega) - H_d(\omega)\right|^2 \le \delta \qquad \text{for } \omega \in \Omega \tag{4.11}$$

where Ω is a frequency region of interest over the positive half of the baseband $[0, \pi]$, then the minimax problem in Eq. (4.7) can be reformulated as

$$\text{minimize } \delta \tag{4.12}$$

$$\text{subject to: } \quad W^2(\omega)\left|H(\omega) - H_d(\omega)\right|^2 \le \delta \qquad \text{for } \omega \in \Omega$$

$$B(z) \ne 0 \quad \text{for } |z| \ge 1$$

we can write

$$W^2(\omega)\left|H(\omega) - H_d(\omega)\right|^2 = \frac{W^2(\omega)}{\left|B(\omega)\right|^2}\left|A(\omega) - B(\omega)H_d(\omega)\right|^2 \tag{4.13}$$

This suggests the following iterative scheme: In the k_{th} iteration, we seek to find polynomials $A_K(z)$ and $B_K(z)$ that solve the constrained optimization problem

$$\text{minimize } \delta \tag{4.14}$$

$$\text{subject to: } \quad \frac{W^2(\omega)}{\left|B_{K-1}(\omega)\right|^2}\left|A(\omega) - B(\omega)H_d(\omega)\right|^2 \le \delta \qquad \text{for } \omega \in \Omega$$

$$B(z) \neq 0 \quad for |z| \geq 1$$

where $B_{K-1}(\omega)$ is obtained in the $(k-1)_{th}$ iteration.

An important difference between the problems in Eqs. (4.12) and (4.14) is that the constraint in Eq. (4.12) is highly nonlinear because of the presence of $B(\omega)$ as the denominator of $H(\omega)$ while the constraint in Eq. (4.14) is a *quadratic* function with respect to the components of a and b and $W^2(\omega)/|B_{K-1}(\omega)|^2$ is a weighting function.

By straightforward manipulations [8], it can be shown that the constrained in Eq. (4.14) is equivalent to

$$\Gamma(\omega) \geq 0 \qquad for\ \omega \in \Omega, \tag{4.15}$$

where

$$\Gamma(\omega) = \begin{bmatrix} \delta & \alpha_1(\omega) & \alpha_2(\omega) \\ \alpha_1(\omega) & 1 & 0 \\ \alpha_2(\omega) & 0 & 1 \end{bmatrix},$$

$$\alpha_1(\omega) = \hat{x}^T c_k - H_{rw}(\omega),$$

$$\alpha_2(\omega) = \hat{x}^T s_k + H_{iw}(\omega),$$

$$\hat{x} = \begin{bmatrix} a \\ b \end{bmatrix}, c_k = \begin{bmatrix} c_w \\ u_w \end{bmatrix}, s_k = \begin{bmatrix} s_w \\ v_w \end{bmatrix},$$

$$w_k = \frac{W(\omega)}{|B_{k-1}(\omega)|},$$

$$c_w = w_k c(\omega), s_w = w_k s(\omega),$$

$$H_{rw}(\omega) = w_k H_r(\omega), H_{iw} = w_k H_i(\omega),$$

$$u_w = w_k[-H_i(\omega)\hat{s}(\omega) - H_r(\omega)\hat{c}(\omega)],$$

and
$$v_w = w_k[-H_i(\omega)\hat{c}(\omega) + H_r(\omega)\hat{s}(\omega)].$$

As for the stability constraint in Eq. (4.14), we note from Sec. 4.2.1 that for a stable filter there exists a $P_{K-1} > 0$ that solves the Lyapunov equation [8]

$$P_{k-1} - D_{k-1}^T P_{k-1} D_{k-1} = I \tag{4.16}$$

where I is the $K \times K$ identity matrix and D_{k-1} is a $K \times K$ matrix of the form in Eq. (4.4) with $-b^T{}_{k-1}$ as its first row. Eq. (4.6) suggests a stability constraint for the digital filter as

$$\begin{bmatrix} P_{k-1}^{-1} & D \\ D^T & P_{k-1} \end{bmatrix} > 0, \tag{4.17}$$

or

$$Q_k = \begin{bmatrix} P_{k-1}^{-1} - \tau I & D \\ D^T & P_{k-1} - \tau I \end{bmatrix} \geq 0, \tag{4.18}$$

Where D is given by Eq. (4.4) and $\tau > 0$ is a scalar that can be used to control the stability margin of the IIR filter. We note that (a) Q_k in Eq. (4.18) depends on D (and hence on $\hat{x}$) affinely; and (b) because of Eq. (4.16), the positive definite matrix P_{K-1} in Eq. (4.18) is *constrained*. Consequently, Eq. (4.18) is a *sufficient* (but not necessary) constraint for stability. However, if the iterative algorithm described above converges, then the matrix sequence $\{D_k\}$ also converges. Since the existence of a $P_{K-1} > 0$ in Eq. (4.16) is a necessary and sufficient condition for the stability of the filter. The LMI constraint in Eq. (4.18) becomes less and less restrictive as the iterations continue.

Combining a discretized version of Eq. (4.15) with the stability constraint in Eq. (4.18), the constrained optimization problem in Eq. (4.14) can now be formulated as

$$\text{minimize} \quad c^T x \tag{4.19}$$

$$\text{subject to} \quad \begin{bmatrix} \Gamma_k & 0 \\ 0 & Q_k \end{bmatrix} \geq 0$$

where

$$x = \begin{bmatrix} \delta \\ \hat{x} \end{bmatrix} = \begin{bmatrix} \delta \\ a \\ b \end{bmatrix}, c = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

and

$$\Gamma_k = diag\{\Gamma(\omega_1), \Gamma(\omega_2), \cdots, \Gamma(\omega_M)\}$$

In the above equation, $\{\omega_i : 1 \leq i \leq M\}$ is a set of frequencies in the range of interest. Since both Γ_k and Q_k depend on variable vector x affinely, the problem in Eq. (4.19) is an SDP problem.

4.3 FIR Low-Pass Filter Design by LMI [8]

In this section we designed FIR Filter using a minimax method based on SDP. In fact, the SDP approach can be used to design FIR filters with arbitrary amplitude and phase responses including certain types of filters that cannot be designed with the another methods such as low-delay FIR filters with approximately constant passband group delay [8].

Consider an FIR filter of order *N* characterized by the general transfer function

$$H(z) = \sum_{n=0}^{N} b_n z^{-n} \tag{4.20}$$

The frequency response of such a filter can be expressed as

$$H(\omega) = \sum_{n=0}^{N} b_n e^{-jn\omega} = b^T c(\omega) - jb^T s(\omega). \tag{4.21}$$

Where $c(\omega)$ and $s(\omega)$ are given by Eqs. (4.10), and $b = [b_0 \; b_1 ...... b_N]^T$. Let be $H_d(\omega)$ the desired frequency response and assume a normalized sampling frequency of 2π. In a minimax design, we need to find a coefficient vector b that solves the optimization problem

$$\underset{b}{\text{minimize}} \quad \max_{\omega \in \Omega} \left[W(\omega) \left| H(\omega) - H_d(\omega) \right| \right] \tag{4.22}$$

where Ω is a frequency region of interest over the positive half of the baseband $[0, \pi]$, and $W(\omega)$ is a given weighting function.

If δ denotes the upper bound of the squared weighted error in Eq. (4.22), i.e.,

$$W^2(\omega) \left| H(\omega) - H_d(\omega) \right|^2 \le \delta \qquad for\ \omega \in \Omega \tag{4.23}$$

Then the minimax problem in Eq. (4.22) can be reformulated as

$$\text{minimize } \delta \tag{4.24}$$

$$\text{subject to } W^2(\omega) \left| H(\omega) - H_d(\omega) \right|^2 \le \delta \qquad for\ \omega \in \Omega \tag{4.25}$$

Now let $H_r(\omega)$ and $H_i(\omega)$ be the real and imaginary parts of $H_d(\omega)$, respectively. We can write

$$W^2(\omega)\left|H(\omega)-H_d(\omega)\right|^2 = W^2(\omega)\left\{\left[b^T c(\omega)-H_r(\omega)\right]^2+\left[b^T s(\omega)+H_i(\omega)\right]^2\right\}$$

$$=\alpha_1^2(\omega)+\alpha_2^2(\omega) \tag{4.26}$$

Where

$$\alpha_1(\omega)=b^T c_w(\omega)-H_{rw}(\omega)$$
$$\alpha_2(\omega)=b^T s_w(\omega)+H_{iw}(\omega)$$

$$c_w=W(\omega)c(\omega)\,,\, s_w=W(\omega)s(\omega)$$

$$H_{rw}(\omega)=W(\omega)H_r(\omega)\,,\, H_{iw}=W(\omega)H_i(\omega)$$

Using Eq. (4.26), the constraint in Eq. (4.25) becomes

$$\delta-\alpha_1^2(\omega)-\alpha_2^2(\omega)\geq 0 \qquad for\ \omega\in\Omega \tag{4.27}$$

It can be shown that the inequality in Eq. (4.21) holds if and only if

$$D(\omega)=\begin{bmatrix}\delta & \alpha_1(\omega) & \alpha_2(\omega)\\ \alpha_1(\omega) & 1 & 0\\ \alpha_2(\omega) & 0 & 1\end{bmatrix}\geq 0 \qquad for\ \omega\in\Omega \tag{4.28}$$

$D(\omega)$ is positive definite for the frequencies of interest [8]. If we write

$$x=\begin{bmatrix}\delta\\ b\end{bmatrix}=\begin{bmatrix}\delta\\ b_0\\ b_1\\ \vdots\\ b_N\end{bmatrix},\, c=\begin{bmatrix}1\\ 0\\ \vdots\\ 0\end{bmatrix} \tag{4.29}$$

Then matrix $D(\omega)$ is *affine* with respect to x. If $S = \{\omega_i : 1 \le i \le M\} \subset \Omega$ is a set of frequencies which is sufficiently dense on Ω, then a discretized version of Eq. (4.22) is given by

$$F(x) \ge 0 \tag{4.30}$$

Where

$$F(x) = diag\{D(\omega_1), D(\omega_2), \cdots, D(\omega_M)\}$$

and the minimization problem in Eq. (4.24) can be converted into the optimization problem

$$\text{minimize } c^T x \tag{4.31}$$

$$\text{subject to } \quad F(x) \ge 0$$

where

$$c = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

The problem in Eq. (4.31) is similar to that in Eq. (1.2), so this problem is an SDP problem.

We have shown that FIR design with LMI constraints which is minimizing the value of the squared weighted error δ between the designed FIR low pass filter and the desired low pass loop filter can be cast as SDP feasibility problems. In fact, many extensions of the problem can be handled by simply adding a cost function and/or LMI constraints to our SDP formulation. The other problem is finding the optimal order of FIR filter which gives the desired specifications.

Minimum-length FIR design [30]

The length of an FIR filter is a quasi-convex function of its coefficients. Hence, the problem of finding the minimum-length FIR filter given magnitude upper and lower bounds as following:

$$\begin{aligned} &\text{minimize} \;\; G \\ &\text{subject to} \;\; L(\omega_i) \le |H_N(\omega_i)| \le U(\omega_i), \quad i = 1, \cdots, M \end{aligned} \tag{4.32}$$

where $G = N+1$ is the length of an FIR filter, $L(\omega)$ is lower magnitude bound and $U(\omega)$ is upper magnitude bound. The problem (4.32) is quasi-convex and can be solved using bisection on N. We will use the following theorem to solve (4.32) .

Theorem 1 [8]:

Given a discrete-time linear system $(A;B;C;D)$, A stable, $(A;B;C)$ minimal and $D + D^T \ge 0$ the transfer function $H(z) = C(zI - A)^{-1}B + D$ satisfies

$$H(\omega) + H^*(\omega) \ge 0 \quad \textit{for all} \;\; \omega \in [0, 2\pi]$$

if and only if there exists real symmetric matrix P such that the matrix inequality

$$\begin{bmatrix} P - A^T PA & C^T - A^T PB \\ C - B^T PA & D + D^T - B^T PB \end{bmatrix} \ge 0 \tag{4.33}$$

is satisfied. The theorem proof is found in [8]. In order to apply Theorem 1, we would like to define $(A;B;C;D)$ in terms of r such that

$$C(zI - A)^{-1}B + D = \frac{1}{2}r(0) + r(1)z^{-1} + \cdots + r(G-1)z^{-(G-1)}, \tag{4.34}$$

where

$$r(g) = \sum_{k=-\infty}^{\infty} h(k)\, h(k+g), \tag{4.35}$$

where we take $h(k) = 0$ for $k < 0$ or $k > G - 1$. The sequence $r(g)$ is symmetric around $g = 0$, zero for $g \leq -G \; or \; g \geq G \; and \; r(0) \geq 0$. Note that the Fourier transform of $r(g)$,

$$R(\omega) = \sum_{g=-\infty}^{\infty} r(g)\, e^{-j\omega g} = \left|H(\omega)\right|^2, \qquad (4.36)$$

is the power spectrum of $h(g)$ and we use r as our design variables.

An obvious choice is the controllability canonical form:

$$A = \begin{bmatrix} 0 & 0 & \cdots & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad (4.37)$$

$$C = \begin{bmatrix} r(1) & r(2) & \cdots & r(G-1) \end{bmatrix}, \qquad D = \frac{1}{2} r(0).$$

It can be easily checked that ($A;B;C;D$) given by (4.37) satisfies (3.34) and all the hypotheses of Theorem 1. Therefore the existence of r and symmetric P that satisfy the matrix inequality (4.33) is the necessary and sufficient condition for $R(\omega) \geq 0$ $for\; all\; \omega \in [0, \pi]$, by Theorem 1.

Note that (4.33) depends affinely on r and P. Thus we can formulate the SDP feasibility problem:

$$\begin{aligned} &\text{find} \quad && r \in R^n \quad and \;\; P = P^T \in R^{(G-1)\times(G-1)} \\ &\text{subject to} \quad && L^2(\omega_i) \leq R(\omega_i) \leq U^2(\omega_i), \quad \omega_i \in \Omega \end{aligned}$$

$$\begin{bmatrix} P - A^T P A & C^T - A^T P B \\ C - B^T P A & D + D^T - B^T P B \end{bmatrix} \geq 0 \qquad (4.38)$$

with (*A;B;C;D*) given by (4.37). The SDP feasibility problem (4.38) can be cast as an ordinary SDP and solved efficiently. Each iteration of the bisection in (4.32) involves solving an SDP feasibility problem in (4.38).

The next step is the design of digital IIR low pass loop filter using Semi-Definite programming (SDP) utilizing Linear Matrix Inequalities formulation (LMI). We will use an SDP based method for the design of digital IIR low pass loop filter under two LMI constraints: first constraint is minimizing the value of the squared weighted error δ between the designed IIR low pass filter and the desired low pass loop filter (ideal case), and second constraint is to design a stable digital IIR low pass loop filter. We will use the MATLAB software to write SDP formulation of the design problem in equation (4.19) and to solve this equation by using SeDuMi toolbox, and then we use the resultant IIR filter in designing fractional-*N* synthesizer in Figure (4-3) as a loop filter to derive the desired Fixed WiMAX frequency (3.5 – 5.8 GHz). We will compare my design results in [32].

After that, we will design a FIR low pass loop filter using SDP utilizing LMI formulation. We will use an SDP based method for the design of digital FIR low pass loop filter under LMI constraint. This constraint is to minimize the value of the squared weighted error δ between the designed FIR low pass filter and the desired low pass loop filter (ideal case). We will use the MATLAB to write SDP formulation of the design problem in equation (4.25) and to solve this equation by using SeDuMi toolbox, and then we use the resultant FIR filter in designed fractional-*N* synthesizer in Figure (4-3) as a loop filter to derive the desired Fixed WiMAX frequency (3.5 – 5.8 GHz). Finally, we will compare my design results with results in [32].

CHAPTER 5

Results Analysis

In this thesis, we will design the PLL loop filter in a frequency synthesizer taking into consideration various design objectives: small settling time, small overshoot and meeting Fixed WiMAX requirements. We first present the PLL frequency synthesizer model (as in Figure 5-1) utilizing the similar parameters that used in [32], where these parameters are integer value (*N*) and reference frequency. The design process is divided into three steps presented as follows: The first step is to design IIR digital low pass filter using SDP algorithms. Second step is to design FIR digital low pass filter using SDP algorithm. The final step is to simulate the designed filters, discuss the results, and compare them with others. To simulate the designed filters, we constructed simulation module shown in Figure 5-1.

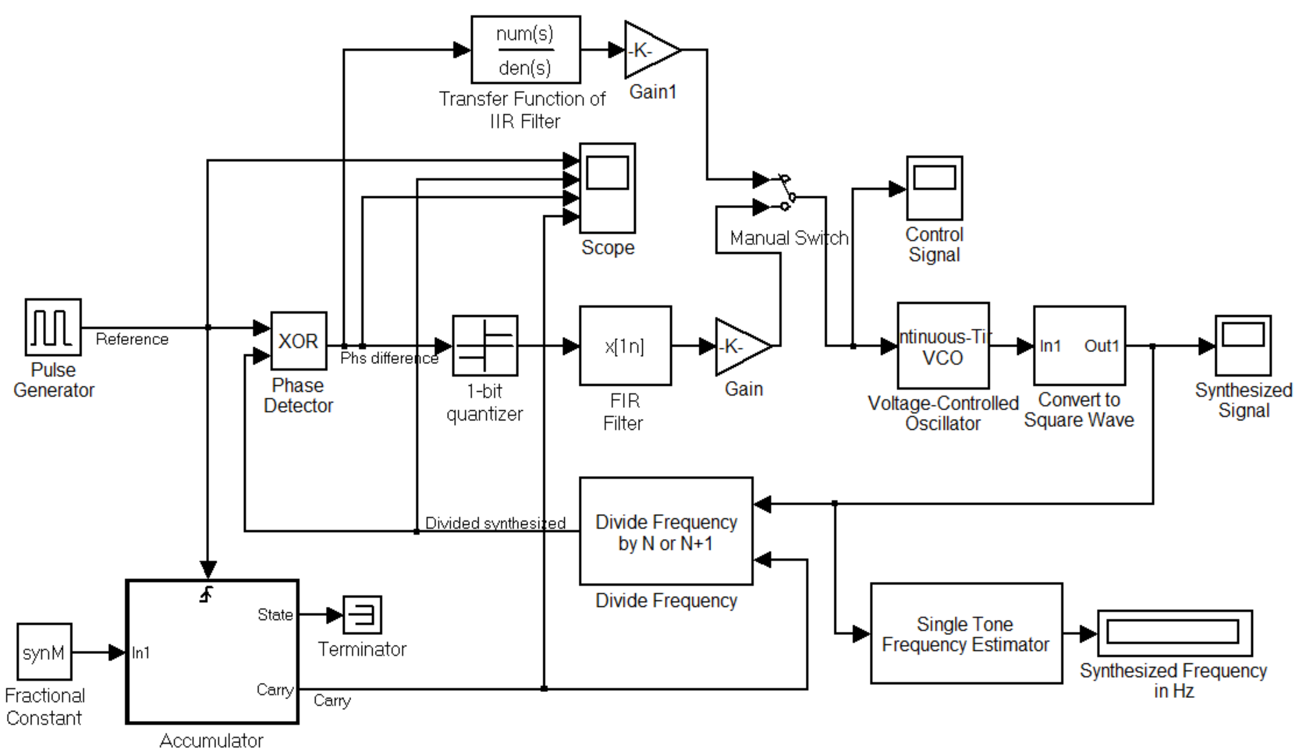

Figure 5-1 PLL Frequency Synthesizer Simulation Model

The simulation module consists of:

1. Reference Frequency: Pulse generator is chosen to produce 20 MHz reference frequency (*synFr*).

2. Two filters (IIR & FIR) are designed and separated by two manual switches as shown in Figure 5-1.

3. Voltage Controlled Oscillator (VCO) with output signal amplitude equal to 1 V, quiescent frequency (*synFq*) equal to 1.511 GHz, and input sensitivity (*synSen*) equal to 10 MHz/V.

4. Phase Detector: XOR type selected.

5. Frequency Divider which produces (*synN* + *synM*) values used to divide the output of VCO. Where *synN* is an integer and *synM* is the fraction.

6. Sigma/Delta Modulator: to produce the required fraction *synM*.

7. The Gain formula $K = K_x * \frac{synFr * synN - synFq}{synSen}$, where K_x = 2.3, 2.4 after the output of IIR filter, FIR filter respectively.

Note that the output synthesized frequency from the previous simulation module (1.533GHz – 2.533GHz) must be multiplied with 2.3 to obtain the required range (3.5GHz – 5.8GHz), see Figure 4-3.

5.1 IIR low pass Filter

IIR low pass Filter can be designed by implementing the procedure described in the previous chapter. In this design we used the same parameters which used in [32] for results comparison.

We choose w_{pass} value compiles with design specifications in table 4-1 as follows:

$$w_{pass} = \Omega_{pass} * \ T \ \ where\ T = \frac{1}{2*2*20MHz} = 0.0125\,\mu s,$$

$$and\ \ \Omega_{pass} = 2\pi(240)\,k\ rad/s.$$

$$\therefore w_{pass} = 2\pi * 240 * 10^{3} * 0.0125 * 10^{-6} = 0.006 * \pi\ rad/s.$$

$w_{stop} = 0.7 * \pi \ rad/s$, chosen after several iterations [32].

We begin IIR filter design process using the following values:

M = 3; % nominator degree

N = 3; % denominator degree

w_{pass} *=0 .006*π; % end of the passband*

w_{stop} *=0 .7*π; % start of the stopband*

δ = 0.0575; % maximum peak passband error

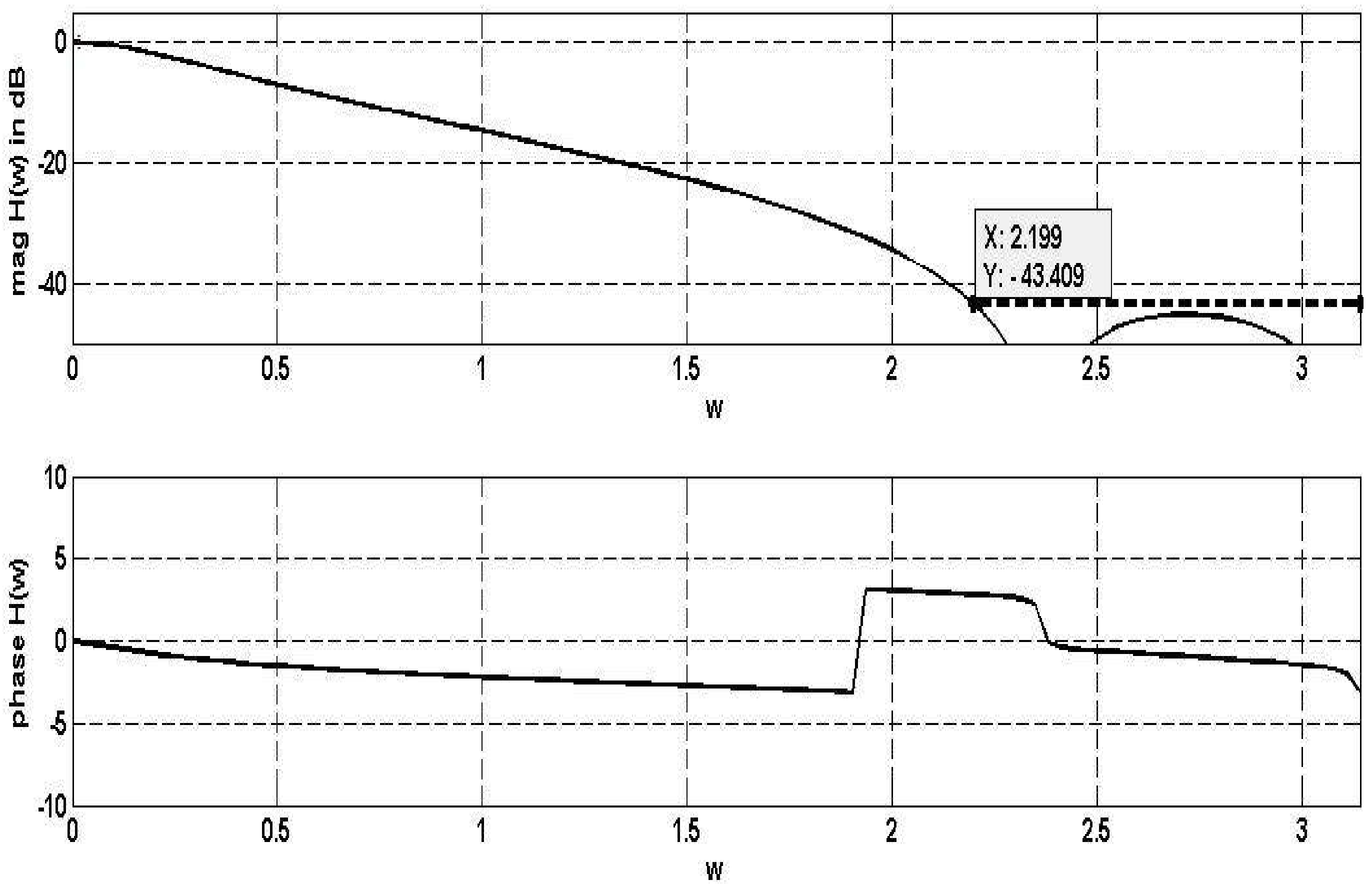

Figure 5-2: 3rd order IIR Filter Magnitude/Phase Response

Note that, the stopband attenuation is not constrained here. Using IIR_by_LMI code listed in the appendix, the resultant IIR filter Transfer Function (transformed to s-domain using the Tustin approximation).

$$T(s) = \frac{-0.0003223s^3 + 6.396*10^6\ s^2 - 9.904*10^{13}\ s + 9.769*10^{23}}{s^3 + 5.114*10^8\ s^2 + 5.45*10^{16}\ s + 9.766*10^{23}}$$

Note, 3rd order filter obtained. The zero and pole locations for IIR filter are:

Zero locations: $s = 1.9837 \times 10^{10}$ *and* $s = (0.0004 \pm 0.0391j) \times 10^{10}$.

Pole locations: $s = -3.7193 \times 10^8, s = -1.1704 \times 10^8$ *and* $s = -0.2244 \times 10^8$.

All poles located in the left half of the *s*-plane so, the system is stable. The filter magnitude/phase response is shown in Figure 5-2.

It is clear from Figure 5-2 that we can summarize the designed IIR filter specifications as follow:

- The maximum pass band ripple = - 0.0179 dB with w_{pass} = 0.006*π *rad/s* .
- Stop band attenuation below - 43.4090 dB with w_{stop} = 0.7*π *rad/s* .

This means that our obtained IIR filter is in complement with passband stage and stopband stage. However, we are encouraged to simulate the obtained IIR filter with Fixed WiMAX simulation block diagram shown in Figure 5-1.

The simulation run properly and the correct output frequency obtained. Figure 5-3 shows the control signal of VCO input using current designed IIR filter. Figure 5-3 shows that:

- The settling time is about 0.5 µs.
- The rise time is very low (0.1 µs).
- The overshoot is eliminated (zero) which agrees with our design specifications listed in Table 4-1.

As a result, we conclude that designing IIR digital filter by SDP programming produced a very fast stable system. Note that, IIR phase consideration is not in our thesis goal. We will discuss and compare these results with [32] in the section 5.3.

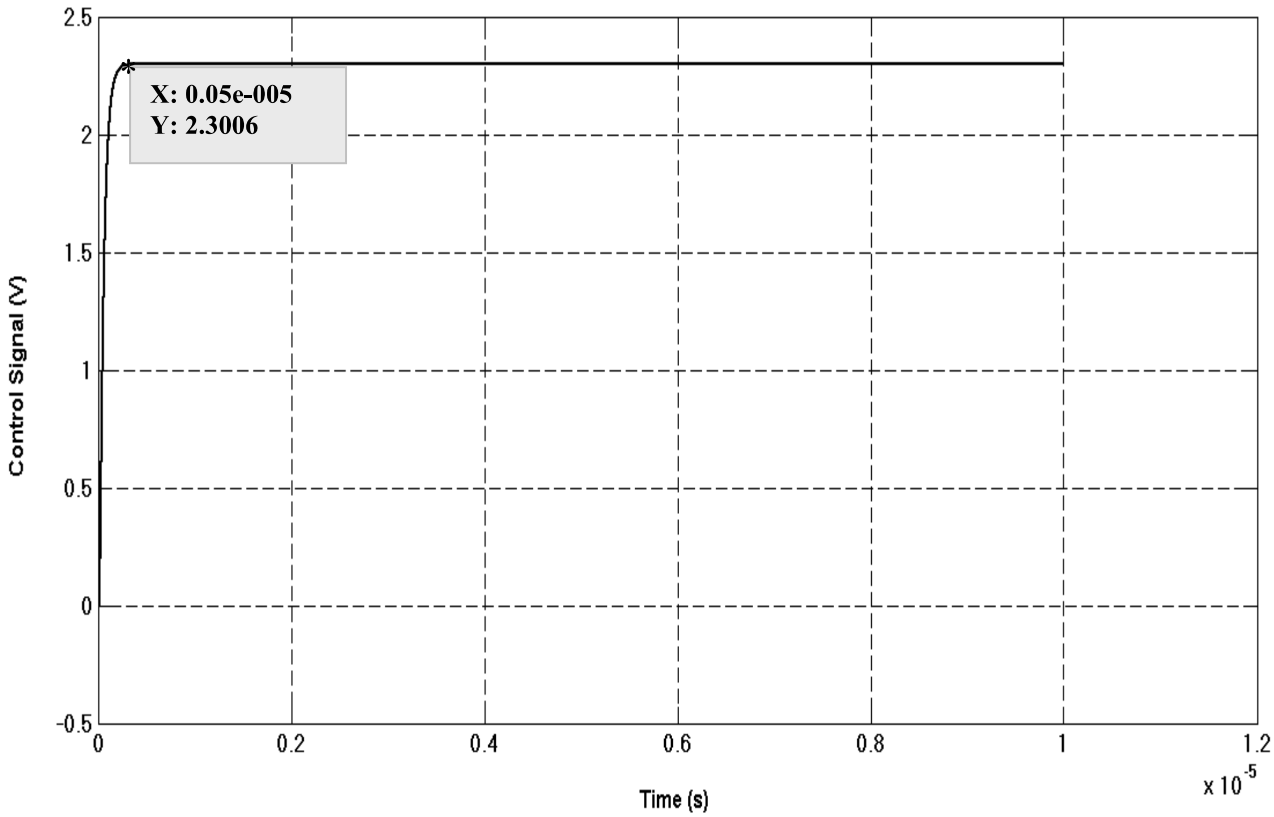

Figure 5-3: The Control Signal of VCO input using Designed IIR Filter

We begin the design process by using SeDuMi toolbox to solve the SDP problem in Eq.(4.19), The name of the toolbox, SeDuMi, stands for Self-Dual Minimization as it implements a self-dual embedding technique for optimization over self-dual homogeneous cones [40]. Presently the official web site for public-domain software SeDuMi can be found in [41], we used in this design the SeDuMi version 1.1R3.

5.2 FIR low-pass Filter

SDP approach can be used to design FIR filters with arbitrary amplitude and phase responses including certain types of filters that cannot be designed with the another methods.

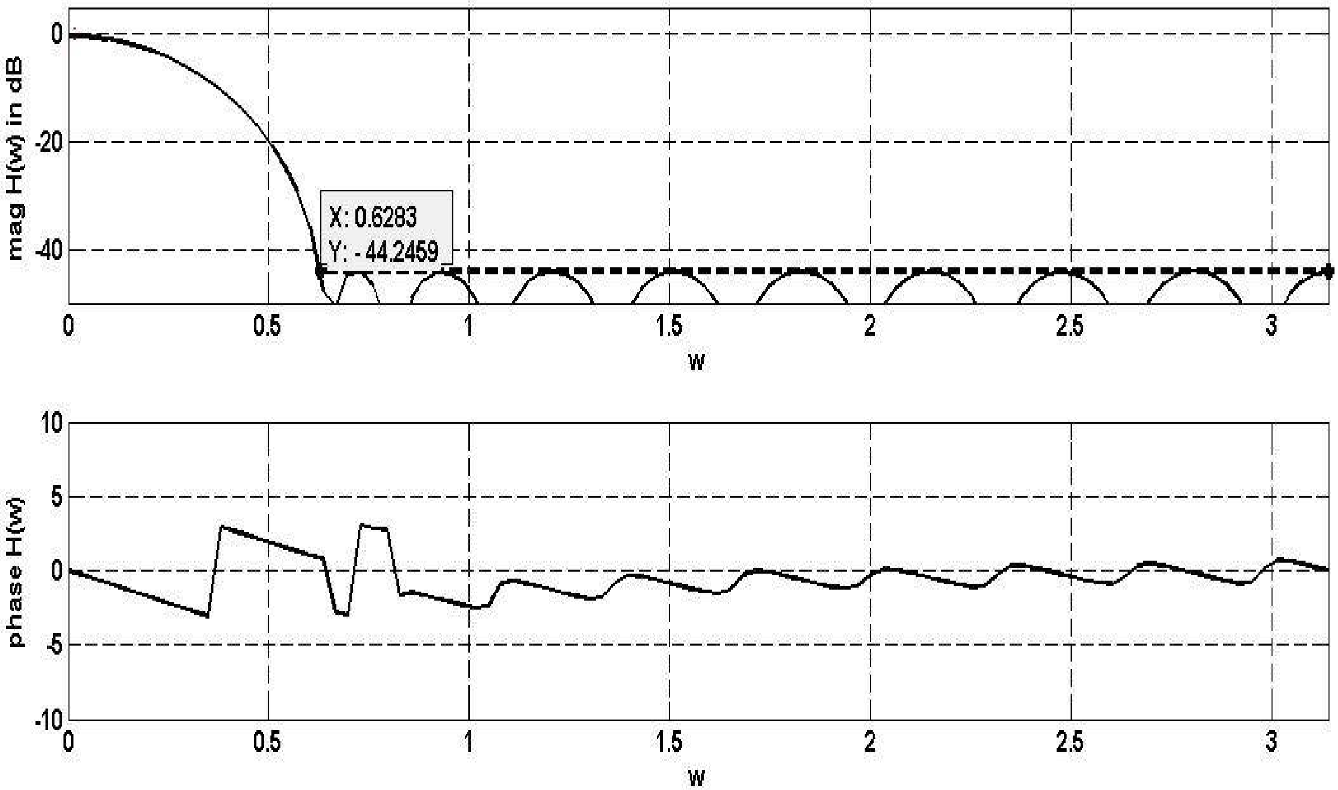

Figure 5-4: FIR Filter Magnitude/Phase Response

We implement the SDP algorithm with these parameters in [32]:

n = 18; % order of filter

w_{pass} *= 0.006*π; % end of the passband* (chosen by same method in section 5.1).

w_{stop} *= 0.2*π; % start of the stopband* (chosen after several iterations [32]).

δ = 0.023; % maximum peak passband error

We begin the design process by using SeDuMi toolbox to solve the SDP problem in Eq.(4.25). Using FIR_by_LMI code listed in the Appendix, the FIR filter magnitude/phase response is shown in Figure 5-4.

It is clear from Figure 5-4 that we can summarize the designed IIR filter specifications as follow:

• The maximum pass band ripple = - 0.3167 dB with w_{pass}= 0.006*π *rad/s* .

• Stop band attenuation below - 44.2459 dB with w_{stop} = 0.2*π *rad/s* .

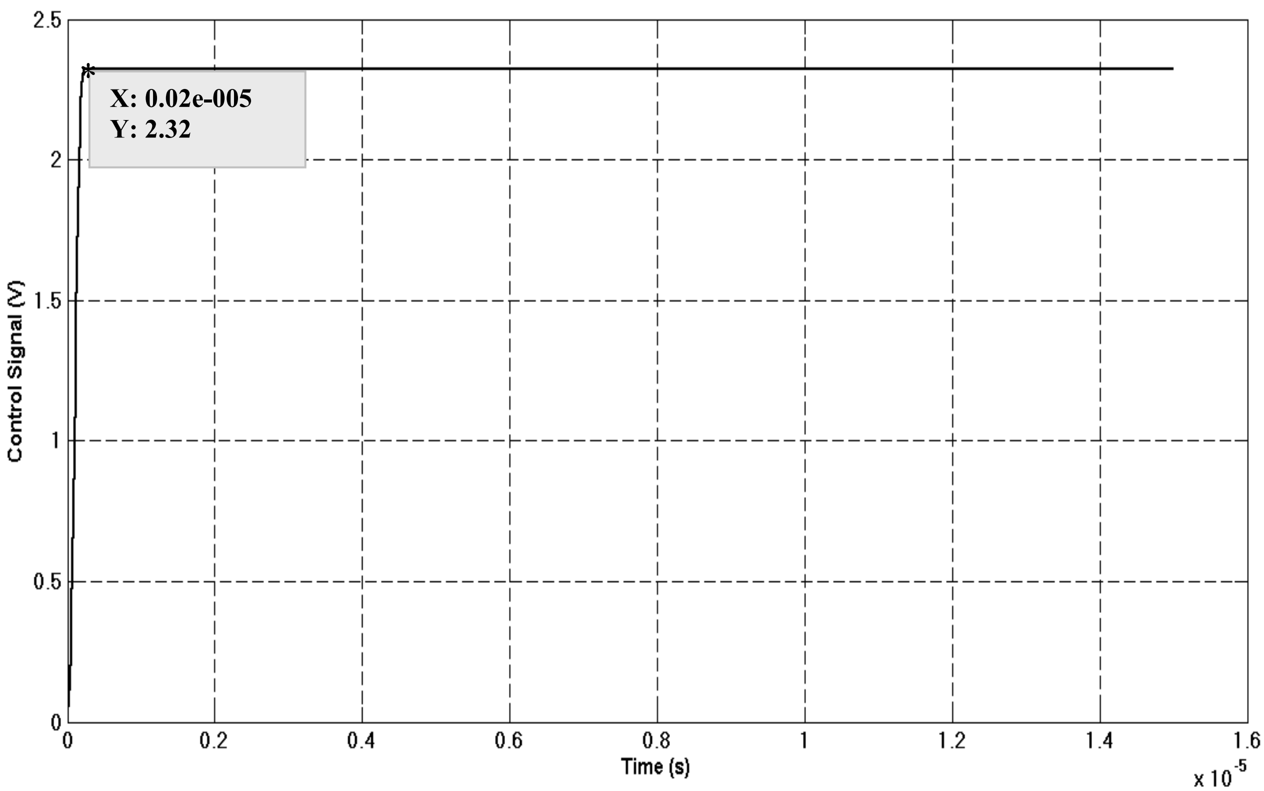

Figure 5-5: The Control Signal of VCO input using Designed FIR Filter

Note that FIR phase does not taken into consideration (FIR phase consideration is not with in our thesis goal).

When we simulate the obtained FIR filter with Fixed WiMAX simulation block diagram shown in Figure 5-1, the simulation work properly and the correct output frequency has been achieved. The control signal using this FIR filter is shown in Figure 5-5 which explore that:

• Overshoot is also eliminated.

• The settling time = 0.2μs.

• The rise time =0.125μs.

From the previous results, the differences between the two control signals that output from two filters (IIR and FIR) are summarized as shown below:

1- Overshoot is eliminated by both filters.
2- Settling time is enhanced (lowered) by FIR filter (0.2 μs for FIR and 0.5 μs for IIR).
3- Rise time is enhanced (lowered) by IIR filter (0.1 μs for IIR and 0.125 μs for FIR).

We have shown that FIR design with minimizing the value of the squared weighted error δ between the designed FIR low pass filter and the desired low pass loop filter (ideal case). In fact, the other problem is finding the minimum-length FIR filter which gives the same results.

Minimum-length FIR design

We implement the minimum-length FIR design algorithm by matlab program (see FIR_min_order MATLAB code listed in the Appendix) with FIR filter parameters but we change the filter order (n = 19):

max_order = 20; % FIR filter length = FIR filter order + 1 = 19 +1=20

w_{pass} = 0.006*π; % passband cutoff freq (in radians)

w_{stop} = 0.2*π; % stopband start freq (in radians)

delta = 0.3167; % max (+/-) passband ripple in dB

atten_level = -44.2459; % stopband attenuation level in dB

When execute FIR_min_order MATLAB code, we found the optimum number of filter taps for given specifications is 19 where the FIR filter order = 19-1=18. This algorithm proof that the order of the FIR filter is optimal for our design specifications. The FIR filter magnitude/phase response is shown in Figure 5-6. The next section discusses results in more detail and compares it with others.

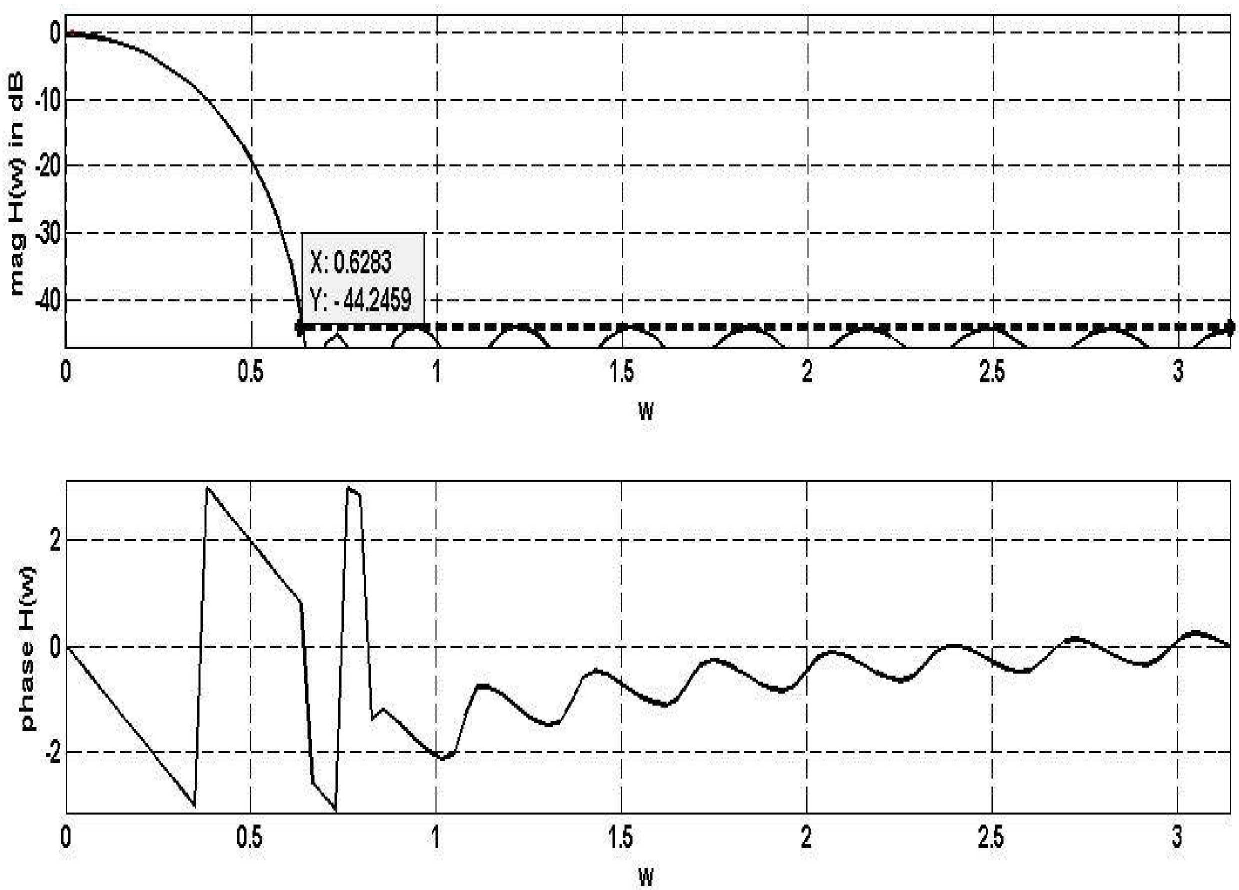

Figure 5-6: FIR_min_order Filter Magnitude/Phase Response

5.3 Discussion

This thesis and Al-Quqa thesis [32] are example for cooperation in the research work. We chose the same research field which is WiMAX technology and we selected the subject of design which is the PLL filter. The differences between this thesis and [32] are the design procedure and the application. As we shown in section 1.1, the WiMAX family of standards addresses two types of usage models: a fixed-usage model (3.5 - 5.8 GHz) which we used and a portable or mobile usage model (2.3 – 2.7 GHz) which is used in [32]. Modern communication like WiMAX technology need to corporate much faster PLL as frequency synthesizer to produce many high frequencies.

The fractional-N frequency synthesizer is used instead of the integer-N frequency synthesizer because the noise was increased by a value equal (10log N) which can be reduced (about 45 dB) by replacing the integer-N with fractional-N frequency synthesizer [32].

As shown in section 4.1, the fractional-N (Figure 4.2) differs than the mobile [32] in terms of the frequency range (3.5 - 5.8 GHz).

More immunity to noise is the main advantage of a digital over an analog filter. For this reason, we used digital a low pass IIR or FIR filter. In [32], the digital IIR low pass filter was designed using Linear Programming (LP) technique and convex optimization software (CVX toolbox) and the digital FIR low pass filter was designed by Linear Programming (LP) technique and Semi-Definite Programming SDP using convex optimization software (CVX toolbox) in two methods. He didn't use Semi-Definite Programming SDP (especially LMI) method to design IIR digital low pass filter. In this thesis, the digital IIR low pass filter and the digital FIR low pass filter are designed using Semi-Definite Programming SDP (SeDuMi toolbox). Using the same IIR filter parameters of [32] (filter order, passband frequency and stopband frequency), we obtained ,as shown in Figure 5.2, the different results in the frequency response compare to [32] results. Table 5.1 summarizes these differences (in the magnitude response):

Table 5-1: Comparison in frequency response of IIR digital filters design

PARAMETER	Al-Quqa IIR Filter (LP)	IIR Filter design (SDP)
Filter order	3	3
Passband frequency(rad/s)	0.006π	0.006π
Stopband frequency(rad/s)	0.7π	0.7π
maximum pass band ripple (dB)	1	0.0179
Stop band attenuation (dB)	59.53	43.41

Note that, IIR phase was not part of the consideration in this thesis; this comparison has been done only in magnitude response. From Table 5.1 with the similar filter order, passband frequency and stopband frequency for two IIR filters, the proposed method in this thesis of minimizing the maximum passband ripple in dB was lower (0.0179 dB) than method in [32] (1 dB). In this design the maximum stopband attenuation (43.41 dB) is lower than value in [32] (59.53 dB). The differences in the frequency magnitude response derived from the differences in the design methods. The proposed filter in frequency synthesizer simulation model is similar to the one used in [32] to have the similar frequency (1.533 GHz). This frequency used in Fractional-*N* Synthesizer, Figure (4.3), to produced the Fixed WiMAX frequencies is in the range of (3.5 -5.8 GHz).

Table 5.2 summarized the differences between this design and [32] in the control signal response for IIR digital filter:

Table 5-2: Comparison in control signal response of IIR digital filters design

PARAMETER	Al-Quqa IIR Filter (LP)	IIR Filter design (SDP)
Settling Time (μs)	4	≈ 0.5
Rise Time (μs)	≈ 1.35	0.1
Maximum Overshoot (%)	zero	zero

From Table 5.2, we conclude that my IIR filter design gave better result with regard to all specification of control signal, where we have the faster system and more stable than system in [32].

We designed the digital FIR low pass filter using SDP (LMI) method using SeDuMi toolbox. In [32], he used two methods to design the FIR digital low pass filter. The first method was using Linear Programming (LP) software and the second method was using Semi-Definite Programming SDP software. He concluded that using FIR digital low pass filter designed with LP will improve the transient behavior of the overall system. Much better transient performance can be achieved with FIR low pass filter designed using SDP. In this thesis, the digital FIR low pass filter is designed by Semi-Definite Programming SDP using SeDuMi toolbox.

When we used the same FIR filter parameters that used in [32] (filter order, passband frequency and stopband frequency), we obtained, as shown in Figure 5.4, the different results in the frequency response compare with [32]. Table 5.3 summarizes these results (in the magnitude response):

Table 5-3: Comparison in frequency response of FIR digital filters design

PARAMETER	Al-Quqa FIR Filter (SDP)	FIR Filter design (SDP)
Filter length	19	19
Passband frequency(rad/s)	0.006π	0.006π
Stopband frequency(rad/s)	0.7π	0.7π
maximum pass band ripple (dB)	0.4	0.3167
Stop band attenuation (dB)	44.5	44.2459

Note that, FIR phase was not considered here, this comparison has be done only in magnitude response. From Table 5.3 with the similar filter order, passband frequency and stopband frequency for both FIR filters, the proposed method of minimizing the maximum passband ripple in dB was lower (0.3167 dB) than method in [32] (0.4 dB). In this design the maximum stopband attenuation (44.2459 dB) is lower than value in [32] (44.5 dB). This design and the design in [32] used the same basic method (SDP) but simulated them using toolbox software (SeDuMi and CVX, respectively).

The proposed filter in frequency synthesizer simulation model is similar to the one used in [32] to have the similar frequency (1.533 GHz). This frequency used in fractional-*N* synthesizer, figure (4.3), to produce the fixed WiMAX frequencies is in the range of (3.5 - 5.8 GHz).

Figure 5.5 and Table 5.4 summarizes the differences between this design and design in [32] about the control signal response for the FIR digital filter:

Table 5-4: Comparison in control signal response of FIR digital filters design

PARAMETER	Al-Quqa FIR Filter (SDP by CVX)	FIR Filter design (SDP by SeDuMi)
Settling Time (μs)	0.4998	≈ 0.2
Rise Time (μs)	≈ 0.25	0.125
Maximum Overshoot (%)	zero	zero

From Table 5-4, we conclude that FIR filter design gave better results in all specification of control signal, where we have the faster system and more stable than system in [32].

We can summarize the previous results as:

- Frequency synthesizer model, as shown in Figure 5.1, was used in Mobile and Fixed WiMax technologies to obtain the same basic frequency (1.533 GHz).
- The basic frequency (1.533 GHz) used in fractional-*N* synthesizer, as shown in Figure 4.3, to obtain the Fixed WiMAX frequency range (3.5 – 5.8 GHz).
- The work concept of fractional-*N* synthesizer in both Fixed and Mobile WiMAX technologies is similar, but the specifications of the

components of fractional-N synthesizer for each technology are different, because the frequency ranges are different.

- Designing the IIR filter using SDP method minimizing the maximum passband ripple in dB was lower (0.0179 dB) than LP method (1 dB).
- The IIR filter which was designed using SDP method gave better result with regard to all specifications of control signal, where we have the faster system and more stable than the system with the IIR filter which was designed using LP method.
- The proposed method of minimizing the maximum passband ripple of FIR filter in dB was lower (0.3167 dB) than Al-Quqa method (0.4 dB).
- The FIR filter which was designed using SDP method and simulated using toolbox software (SeDuMi) gave better result with regard to all specifications of control signal, where we have the faster system and more stable than the system which have FIR filter which designed using SDP method and simulated using toolbox software (CVX).
- The minimum-length FIR filter algorithm was used to proof that the order of the FIR filter (order = 18) which was designed using SDP method and simulated using toolbox software (SeDuMi) is optimal for our design specifications.

CHAPTER 6

Conclusion and Future Work

Phase locked loop is an interesting topic for the research, because of its usage in many applications (electrical - control – communication …etc).

It covers many discipline of electrical engineering such as communication theory, control theory, signal analysis, design with transistors and op amps, digital circuit design and nonlinear analysis.

6.1 Conclusion

A new loop filter design method for frequency synthesizer was introduced taking into consideration various design objectives: small settling time, small overshoot and meeting Fixed WiMAX requirements. IIR and FIR digital low pass filters were designed using semi-definite programming.

The PLL frequency synthesizer model was presented utilizing similar parameters that used in [32], where these parameters are integer value (N) and reference frequency.

The SDP based method for the design of the digital IIR low pass loop filter was used under two LMI constraints: first constraint was minimizing the value of the squared weighted error δ between the designed IIR low pass filter and the desired low pass loop filter (ideal case). The second constraint was to design a stable digital IIR low-pass loop filter. The MATLAB was used to write SDP formulation of the design problem in equation and solved this equation by using SeDuMi toolbox (Self-Dual Minimization), and then we derived the desired Fixed WiMAX frequency (3.5 – 5.8 GHz) utilizing the resultant IIR filter in designed fractional-N synthesizer as a loop filter. Finally, the design results were compared with results in [32].

The results showed that the IIR filter which was designed using SDP method minimizing the maximum passband ripple in dB was lower than LP method. SDP method gave better result with regard to all specifications of control signal, where we have the faster system and more stable than the system which have IIR filter which designed using LP method.

The SDP based method for the design of the digital FIR low pass loop filter was used under LMI constraint. This constraint was minimizing the value of the squared weighted error δ between the designed FIR low pass filter and the desired low pass loop filter (ideal case). The Matlab was used to write SDP formulation of the design problem in equation and solved this equation by using SeDuMi toolbox (Self-Dual Minimization), and then the resultant FIR filter was used in designed fractional-N synthesizer as a loop filter to derive the desired Fixed WiMAX frequency (3.5 – 5.8 GHz). Finally, the design results were compared with results in [32].

Simulations showed that the FIR filter which was designed using SDP method and simulated using toolbox software (SeDuMi) minimizing the maximum passband ripple of FIR filter in dB was lower than the method in [32]. The proposed method gave better result with regard to all specifications of control signal, where we have the faster system and more stable than the system with FIR filter which was designed using SDP method and simulated using toolbox software (CVX). The minimum-length FIR filter algorithm was used to proof that the order of the FIR filter which was designed using SDP method and simulated using toolbox software (SeDuMi) is optimal for our design specifications.

6.2 Future work

We can summarize our suggestions for future works as follows:

- We recommend design of IIR and FIR low pass digital filters to extend the constrained region to include the transition band between pass band and stop band, and to take into consideration the filter phase.
- I used the ideal low pass filter as the desired filter; it can be taking into consideration the time delay in the filter design.
- Semi-Definite Problem can be solved utilizing many optimizing software packages, in [32] was used toolbox software (CVX) and I used toolbox software (SeDuMi); however, it can be combined different optimizing software packages to solve SDP problem.

CHAPTER 7

References

[1] J. Andrews, A. Ghosh, and R. Muhamed, *Fundamentals of WiMAX Understanding Broadband Wireless Networking*, T. Rappaport, N.J.: Pearson Education, 2007.

[2] J. Ahmad, B. Garrison, J. Gruen, C. Kelly, and H. Pankey, "4G Wireless Systems: Next-Generation Wireless Working Group", [Online document], May 2, 2003, Available: http://ckdake.com/portfolio/papers.html.

[3] P. Ames and J. Gabor, "The Evolution of third-generation cellular standards", *Intel Technology Journal*, vol. 4, issue 2, pp. 1-6, 2000.

[4] WiMax Forum, “WiMAX Forum Overview”, [Online document], Retrieved Octobar 31, 2008, Available: http://www.Wimaxforum.org/home/.

[5] G. Hsieh and J. Huang, "Phase-Locked Loop Techniques- A Survey,” *IEEE Trans on Industrial Electronics*, vol. 43, no.6, pp.609-615, 1996.

[6] D. Abramovich, “Phase-Locked Loop: a control centric tutorial,” in *Proceedings of the American Control Conference*, vol. 1, Anchorage, AK, 2002, pp. 1–15.

[7] S. Boyd, L. El Ghaoui, E. Feron, & V. Balakrishnan, Linear *Matrix Inequalities in System and Control Theory.* Philadelphia: SIAM, 1994.

[8] A. Antoniou and W.-S. Lu, *Optimization: Algorithms and Applications*. New York: Springer, 2007.

[9] L. Vandenberghe, & S. Boyd, “Semidefinite programming”, *Society for Industrial and Applied Mathematics*, vol. 38, no. 1, pp. 49-95, 1996.

[10] R. Best, *Phase-Locked Loops: Design, Simulation, and applications, Sixth Edition*, New York: McGraw Hill International, 2007.

[11] D. Abramovitch, "Analysis and design of a third order phase-lock loop," in *Pro. of the IEEE Military Communications Conference,* vol. 2, 1988, pp. 455-459.

[12] D. Abramovitch, "Lyapunov Redesign of analog phase-lock loops," *IEEE Trans. on Communication*, vol. 38, no. 12, pp. 2197-2202, 1990.

[13] M. Abbas-Turki, G. Duc, & B. Clement, "LMI formulation for designing controllers according to time response and stability margin constraints", *in Proc. 44th IEEE Conference on Decision and Control 2005,* Seville, Splain ,France, 2005, pp. 5740 - 5745.

[14] V. Suplin and U. Shaked, "Mixed H_∞ / H_2 Design of Digital phase-locked loops with polytopic type uncertainties," in *Proc. IEEE Conf. Decision and Control* , Sydney, Australia, 2000, pp.7803-6638-7.

[15] 0. Yaniv and D. Raphaeli, "Near-optimal PLL design for decision feedback carrier and timing recovery," *IEEE Trans. on Communication*, vol. 49, no. 9, pp. 1669 - 1678, 2001.

[16] N. Eva Wu," Analog Phaselock Loop Design Using Popov Criterion", in *Proc. of the Amencan Control Conference,* Anchorage. *AK* May 8-10, 2002, pp. 16-18.

[17] M. Mansuri and C. Yang," Jitter Optimization Based on Phase-Locked Loop Design Parameters," *IEEE Journal of solid-state circuits*, vol. 37, no. 11, pp. 1375-1382, 2002.

[18] D. Colleran, C. Portmannt, A. Hassibi, C. Crusius, S. Mohan, S. Boyd, T. Leet and M. Hershenson, "Optimization of Phase-Locked Loop Circuits via Geometric Programming," in *IEEE Custom Integrated Circuits Conference,* 2003, pp.377-380.

[19] A. Fahim and M. Elmasry, "A Fast Lock Digital Phase-Locked-Loop Architecture for Wireless Applications," *IEEE Transaction on Circuits and Systems*, vol. 50, no. 2, pp. 63-72, 2003.

[20] M. Ghartemani, H. Karimi, and M. Iravani, "A Magnitude/Phase-Locked Loop System Based on Estimation of Frequency and In-Phase/Quadrature-Phase Amplitudes," *IEEE Transaction on Industrial Electronics*, vol. 51, no. 2, pp.511-517, 2004.

[21] W. Mao, H. Tsao, and F. Chang, "Intelligent GPS receiver for robust carrier phase tracking in kinematic environments," in *IEE Proceedings Radar-Sonar and Navigation*, 2004, pp.171-180.

[22] Y. Chou, W. Mao, Y. Chen and F. Chang, "A Novel Loop Filter Design for Phase-Locked Loops", *IEEE International Conference on Systems Man and Cybernetics,* Taipei, Taiwan, October 8-11, 2006, pp. 2932-2938.

[23] K. Kundert, "Predicting the phase noise and jitter of PLL-based frequency synthesizer," [Online document], 2006, Available: http://www.designers-guide.org/.

[24] C. Barrett, "Fractional/Integer-N PLL Basics," [Online document], 1999, Available: http://focus.ti.com/lit/an/swra029/swra029.pdf.

[25] C. Lau, & M. Perrott, "Fractional-N frequency synthesizer design at the transfer function level using a direct closed loop realization algorithm", in *Proceedings Design Automation Conference*, New York, 2003, pp. 526 - 531.

[26] M. Kozak, & E. Friedman, "Design and simulation of fractional-N PLL frequency synthesizers", in *Proceedings of the 2004 International Symposium on Circuits and Systems*, 2004, pp. IV- 780-3.

[27] T. Wang, T. Chiou, and S. Lall., "Nonlinear Phase-Locked Loop Design using Semidefinite Programming", in *Proceedings of the Mediterranean Conference on Control and Automation* , 2008, pp. 1640 – 1645.

[28] N. Long, K. Hyunsoo, E. Yang, & K. Hansil, "LMI approach to H_∞optimal filter design for GPS receiver tracking loop", in *IEEE International Symposium on Electrical & Electronics Engineering.* HCM City, Vietnam, 2007.

[29] W. Al-Baroudi, "*Digital Filter Design using LMI Based Techniques",* Thesis. Faculty of the Graduate Studies, King Fahd University of Petroleum & Minerals, 1997.

[30] S. Wu, S. Boyd, & L. Vandenberghe, "FIR filter design via semidefinite programming and spectral factorization", in *Proceedings of the 35th IEEE Decision and Control*, 1996, pp. 271 – 276.

[31] L. Rabiner, N. Graham, & H. Helms, "Linear programming design of IIR digital filters with arbitrary magnitude function", *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. *22*, no. 2, pp. 117 – 123, 1974.

[32] A. Al-Quqa, "*Optimal PLL Loop Filter Design for Mobile WiMax Via LM",* Thesis. Faculty of the Graduate Studies, Islamic University of Gaza, 2009.

[33] A. Antoniou, *Digital Signal Processing: Signals, Systems, and Filters*, New York, McGraw-Hill, 2005.

[34] W. Lu., "Design of Stable Minimax IIR Digital Filters Using Semidefinitte Programming", ISCAS 2000, Geneva, Switzerland, May 28-31, 2000, pp. I-355-358.

[35] Deepak Pareek, *The Business of WiMAX*, England, John Wiley & Sons Ltd, 2006.

[36] J. Andrews, A. Ghosh and R. Muhamed, *Fundamentals of WiMAX: Understanding Broadband Wireless Networking*, Pearson Education, Inc., 2007.

[37] D. Sweeney**,** *WiMax Operator's Manual: Building 802.16 Wireless Networks*, United States, après, 2006.

[38] F. Ohrtman, "Fixed vs. Mobile WiMAX: Mobility Monomania", Distributed by Mind Commerce, July 30, 2007.

[39] Telecoms of Korea, "Mobile WiMAX Handbook", [Online document], 2008, Available: http://www.telecomskorea.com.

[40] W. Lu**,** “Use SeDuMi to Solve LP, SDP and SCOP Problems: Remarks and Examples”, Dept. of Electrical and Computer Engineering, University of Victoria, January 29, 2007.

[41] McMaster University, Official web site for public-domain software SeDuMi, [Online], 2003, Available: http://sedumi.mcmaster.ca/.

CHAPTER 8

Appendix

8.1 IIR_by_LMI MATLAB Code

```
% Minimax design of IIR digital filters by SDP(semidefinite programming)
% using SeDuMi(self-dual minimization)toolbox
% introduced by Fady El-Batta
clear;
clc;
N = 3;                   % order of nominator
K = 3;                   % order of dinominator
M =100;                  % N.O. equally-spaced sample frequencies
t = 1e-5;                % taw is scalar used to control stability margin
j = sqrt(-1);
wpass = 0.006*pi;        % end of the passband
wstop = 0.7*pi;          % start of the stopband
W     = 6e-4;            % weighting fuction W(w)
delta = 0.0575;          % maximum peak passband error(ripple = 1 dB)

%%%%%%%%%%%%%%%%%%% Initial Values %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 wn =  (wpass+wstop)/(2*pi);
 B   = fir1(N,wn);       % Hamming-window based
 a   = B';               % coefficients vector of nominator
 b   = zeros(K,1);       % coefficients vector of dinominator
 xx = [a;b];             % initial value of xx
 x   = [delta;a;b];      % initial value of x

%%%%%%%%%%%%%%%% constraint of stability %%%%%%%%%%%%%%%%%%%%%%%
v     = ones(1,K-1);
I     = eye(K);
D     = [-1*b';diag(v,0) [zeros(1,K-1)]']; %Dk-1
P     = dlyap(D',I);     %Pk-1
Pin   = P^-1;
Qk    = [ Pin-t*eye(K)    D;D'     P-t*eye(K)];

%%%%%%%%%%%%%%%%%% Discretization %%%%%%%%%%%%%%%%%%%%%%%%%%
w1   = linspace(0,wpass,M/2);
w2   = linspace(wstop,pi,M/2);
w    = [w1 w2];                     % omega

%%%%%%%%%%%%%%%%%%% SDP problem %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% formulate the SDP problem

gamak = [];
 for i = 1:1:M
      wi     = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi); % desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
```

```
          Hi     = 0;% imaginary part of desired frequency response
          Hd     = complex(Hr,Hi); % desired frequency response
          end
     c      = [1 cos([1:N]*wi)]';
     s      = [0 sin([1:N]*wi)]';
     c1     = [cos([1:K]*wi)]';
     s1     = [sin([1:K]*wi)]';
     Aw     = (a' * c) - j*(a'* s);
     Bw     = 1 + (b' * c1) - j*(b' * s1);
     wk     = W/abs(Bw);
     Hrw    = wk*Hr;
     cw     = wk*c;
     uw     = wk*[-Hi*s1-Hr*c1];
     ck     = [cw;uw];
     a1     = xx'*ck - Hrw;
     vw     = wk*[-Hi*c1+Hr*s1];
     sw     = wk*s;
     sk     = [sw;vw];
     Hiw    = wk*Hi;
     a2     = xx'*sk + Hiw;
     gamaw = [delta a1 a2;a1 1 0;a2 0 1];
     gamax = [gamaw zeros(3,3*(M-1))];
     gamax = circshift(gamax,[0 3*(i-1)]);
     gamak = [gamak; gamax];

 end
 z      = zeros (3*M,2*K);
 F      =  [ gamak                              z; ...
           z'                                 Qk] ;

%%%%%%%%%%%%%%%%%% Using SeDuMi %%%%%%%%%%%%%%%%%%%%%%%%%
%%%%F = A0+y(1)A1+y(2)A2+......+y(7)A7+y(8)A8 >= 0
%%%%%%% calculate A0
gamak = [];
 for i = 1:1:M
      wi     = w(1,i);
          if      (0 <= wi) && (wi <= wpass)
          Hr     = cos(0*wi);% real part of desired frequency response
          Hi     = sin(0*wi);% imaginary part of desired frequency response
          Hd     = complex(Hr,Hi);% desired frequency response
          elseif (wstop <= wi) && (wi <= pi)
          Hr     = 0;% real part of desired frequency response
          Hi     = 0;% imaginary part of desired frequency response
          Hd     = complex(Hr,Hi);% desired frequency response
          end
     c1     = [cos([1:K]*wi)]';
     s1     = [sin([1:K]*wi)]';
     Bw     = 1 + (b' * c1) - j*(b' * s1);
     wk     = W/abs(Bw);
     Hrw    = wk*Hr;
     a1     = - Hrw;
     Hiw    = wk*Hi;
     a2     = Hiw;
     gamaw = [0 a1 a2;a1 1 0;a2 0 1];
     gamax = [gamaw zeros(3,3*(M-1))];
     gamax = circshift(gamax,[0 3*(i-1)]);
```

```
    gamak = [gamak; gamax];

 end
 z      = zeros (3*M,2*K);
 Qk(1,4) = 0;
 Qk(4,1) =Qk(1,4);
 Qk(1,5) = 0;
 Qk(5,1) =Qk(1,5);
 Qk(1,6) = 0;
 Qk(6,1) =Qk(1,6);
 A0     =  [ gamak                          z; ...
          z'                               Qk] ;
%%%%%%% calculate A1
s   = size(F);
r   = s(1,1);
A1 = zeros(r);
g=1;
for i=1:1:M
    A1(g,g)= 1;
    g = g+3;
end
%%%%%%% calculate A2
gamak = [];
 for i = 1:1:M
      wi     = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
    c      = [1 cos([1:N]*wi)]';
    s      = [0 sin([1:N]*wi)]';
    c1     = [cos([1:K]*wi)]';
    s1     = [sin([1:K]*wi)]';
    Aw     = (a' * c) - j*(a'* s);
    Bw     = 1 + (b' * c1) - j*(b' * s1);
    wk     = W/abs(Bw);
    a1     = wk*c(1);
    a2     = wk*s(1);
    gamaw = [delta a1 a2;a1 1 0;a2 0 1];
    gamax = [gamaw zeros(3,3*(M-1))];
    gamax = circshift(gamax,[0 3*(i-1)]);
    gamak = [gamak; gamax];

 end
  z      = zeros (3*M,2*K);
 Qk     = zeros (2*K,2*K);
 A2      =  [ gamak                          z; ...
          z'                               Qk] ;
for j=1:1:r
    for k=1:1:r
        if j==k
```

```
        A2(j,k)=0;
        end
    end
end
%%%%%%% calculate A3
gamak = [];
 for i = 1:1:M
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
    c      = [1 cos([1:N]*wi)]';
    s      = [0 sin([1:N]*wi)]';
    c1     = [cos([1:K]*wi)]';
    s1     = [sin([1:K]*wi)]';
    Aw     = (a' * c) - j*(a'* s);
    Bw     = 1 + (b' * c1) - j*(b' * s1);
    wk     = W/abs(Bw);
    a1     = wk*c(2);
    a2     = wk*s(2);
    gamaw = [delta a1 a2;a1 1 0;a2 0 1];
    gamax = [gamaw zeros(3,3*(M-1))];
    gamax = circshift(gamax,[0 3*(i-1)]);
    gamak = [gamak; gamax];

 end
  z      = zeros (3*M,2*K);
 Qk     = zeros (2*K,2*K);
 A3      =  [ gamak                            z; ...
          z'                               Qk] ;
for j=1:1:r
    for k=1:1:r
        if j==k
        A3(j,k)=0;
        end
    end
end
%%%%%%% calculate A4
gamak = [];
 for i = 1:1:M
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
```

```
    c      = [1 cos([1:N]*wi)]';
    s      = [0 sin([1:N]*wi)]';
    c1     = [cos([1:K]*wi)]';
    s1     = [sin([1:K]*wi)]';
    Aw     = (a' * c) - j*(a'* s);
    Bw     = 1 + (b' * c1) - j*(b' * s1);
    wk     = W/abs(Bw);
    a1     = wk*c(3);
    a2     = wk*s(3);
    gamaw = [delta a1 a2;a1 1 0;a2 0 1];
    gamax = [gamaw zeros(3,3*(M-1))];
    gamax = circshift(gamax,[0 3*(i-1)]);
    gamak = [gamak; gamax];

 end
  z      = zeros (3*M,2*K);
 Qk     = zeros (2*K,2*K);
 A4      =  [ gamak                              z; ...
         z'                                Qk] ;
for j=1:1:r
    for k=1:1:r
        if j==k
        A4(j,k)=0;
        end
    end
end
%%%%%%% calculate A5
gamak = [];
 for i = 1:1:M
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
    c      = [1 cos([1:N]*wi)]';
    s      = [0 sin([1:N]*wi)]';
    c1     = [cos([1:K]*wi)]';
    s1     = [sin([1:K]*wi)]';
    Aw     = (a' * c) - j*(a'* s);
    Bw     = 1 + (b' * c1) - j*(b' * s1);
    wk     = W/abs(Bw);
    a1     = wk*c(4);
    a2     = wk*s(4);
    gamaw = [delta a1 a2;a1 1 0;a2 0 1];
    gamax = [gamaw zeros(3,3*(M-1))];
    gamax = circshift(gamax,[0 3*(i-1)]);
    gamak = [gamak; gamax];

 end
  z      = zeros (3*M,2*K);
 Qk     = zeros (2*K,2*K);
```

```
 A5      =  [ gamak                              z; ...
              z'                              Qk] ;
for j=1:1:r
    for k=1:1:r
        if j==k
        A5(j,k)=0;
        end
    end
end
%%%%%%% calculate A6
gamak = [];
 for i = 1:1:M
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
    c      = [1 cos([1:N]*wi)]';
    s      = [0 sin([1:N]*wi)]';
    c1     = [cos([1:K]*wi)]';
    s1     = [sin([1:K]*wi)]';
    Aw     = (a' * c) - j*(a'* s);
    Bw     = 1 + (b' * c1) - j*(b' * s1);
    wk     = W/abs(Bw);
    Hrw    = wk*Hr;
    cw     = wk*c;
    uw     = wk*[-Hi*s1-Hr*c1];
    ck     = [cw;uw];
    a1     = uw(1);
    vw     = wk*[-Hi*c1+Hr*s1];
    sw     = wk*s;
    sk     = [sw;vw];
    Hiw    = wk*Hi;
    a2     = vw(1);
    gamaw = [delta a1 a2;a1 1 0;a2 0 1];
    gamax = [gamaw zeros(3,3*(M-1))];
    gamax = circshift(gamax,[0 3*(i-1)]);
    gamak = [gamak; gamax];

 end
 z      = zeros (3*M,2*K);
 Qk     = zeros (2*K,2*K);
 Qk(1,4)  = -1;
 Qk(4,1)  =Qk(1,4);
 A6      =  [ gamak                              z; ...
              z'                              Qk] ;
 for j=1:1:r
    for k=1:1:r
        if j==k
        A6(j,k)=0;
        end
```

```
    end
 end
%%%%%%% calculate A7
gamak = [];
 for i = 1:1:M
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr    = cos(0*wi);% real part of desired frequency response
         Hi    = sin(0*wi);% imaginary part of desired frequency response
         Hd    = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr    = 0;% real part of desired frequency response
         Hi    = 0;% imaginary part of desired frequency response
         Hd    = complex(Hr,Hi);% desired frequency response
         end
    c      = [1 cos([1:N]*wi)]';
    s      = [0 sin([1:N]*wi)]';
    c1     = [cos([1:K]*wi)]';
    s1     = [sin([1:K]*wi)]';
    Aw     = (a' * c) - j*(a'* s);
    Bw     = 1 + (b' * c1) - j*(b' * s1);
    wk     = W/abs(Bw);
    Hrw    = wk*Hr;
    cw     = wk*c;
    uw     = wk*[-Hi*s1-Hr*c1];
    ck     = [cw;uw];
    a1     = uw(2);
    vw     = wk*[-Hi*c1+Hr*s1];
    sw     = wk*s;
    sk     = [sw;vw];
    Hiw    = wk*Hi;
    a2     = vw(2);
    gamaw = [delta a1 a2;a1 1 0;a2 0 1];
    gamax = [gamaw zeros(3,3*(M-1))];
    gamax = circshift(gamax,[0 3*(i-1)]);
    gamak = [gamak; gamax];

 end
 z      = zeros (3*M,2*K);
 Qk     = zeros (2*K,2*K);
 Qk(1,5)  = -1;
 Qk(5,1)  =Qk(1,5);
 A7      =  [ gamak                          z; ...
          z'                              Qk] ;
 for j=1:1:r
    for k=1:1:r
        if j==k
        A7(j,k)=0;
        end
    end
 end
 %%%%%%% calculate A8
gamak = [];
 for i = 1:1:M
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
```

```
        Hr    = cos(0*wi);% real part of desired frequency response
        Hi    = sin(0*wi);% imaginary part of desired frequency response
        Hd    = complex(Hr,Hi);% desired frequency response
        elseif (wstop <= wi) && (wi <= pi)
        Hr    = 0;% real part of desired frequency response
        Hi    = 0;% imaginary part of desired frequency response
        Hd    = complex(Hr,Hi);% desired frequency response
        end
   c      = [1 cos([1:N]*wi)]';
   s      = [0 sin([1:N]*wi)]';
   c1     = [cos([1:K]*wi)]';
   s1     = [sin([1:K]*wi)]';
   Aw     = (a' * c) - j*(a'* s);
   Bw     = 1 + (b' * c1) - j*(b' * s1);
   wk     = W/abs(Bw);
   Hrw    = wk*Hr;
   cw     = wk*c;
   uw     = wk*[-Hi*s1-Hr*c1];
   ck     = [cw;uw];
   a1     = uw(3);
   vw     = wk*[-Hi*c1+Hr*s1];
   sw     = wk*s;
   sk     = [sw;vw];
   Hiw    = wk*Hi;
   a2     = vw(3);
   gamaw = [delta a1 a2;a1 1 0;a2 0 1];
   gamax = [gamaw zeros(3,3*(M-1))];
   gamax = circshift(gamax,[0 3*(i-1)]);
   gamak = [gamak; gamax];

end
z      = zeros (3*M,2*K);
Qk     = zeros (2*K,2*K);
Qk(1,6)  = -1;
Qk(6,1) =Qk(1,6);
A8      =  [ gamak                              z; ...
         z'                               Qk] ;
for j=1:1:r
   for k=1:1:r
       if j==k
       A8(j,k)=0;
       end
   end
end

 %%%%%%%%%%%%%%%%%SeDuMi function
p   = length([1 0 0 0 0 0 0 0]);
bt   = -[1 0 0 0 0 0 0 0];
ct   = vec(A0);
K.s = size(A0,1);
At(:,1) = -vec(A1);
At(:,2) = -vec(A2);
At(:,3) = -vec(A3);
At(:,4) = -vec(A4);
At(:,5) = -vec(A5);
At(:,6) = -vec(A6);
```

```
At(:,7) = -vec(A7);
At(:,8) = -vec(A8);
[x, y, info] = sedumi(At,bt,ct,K);
y
%%%%%%%%%%%%%%%%%%%%%%% Plot %%%%%%%%%%%%%%%%%%%%%%%%%%%
a=y(2:5);          % coefficients vector of nominator
b=y(6:8);          % coefficients vector of dinominator
N =3;              % order of nominator
K = 3;             % order of dinominator
M =100;            % N.O. equally-spaced sample frequencies
j = sqrt(-1);
wpass = 0.006*pi;      % end of the passband
wstop = 0.7*pi;        % start of the stopband
w  = linspace(0,pi,M); % omega
for i = 1:1:M
    wi    = w(1,i);
    c     = [1 cos([1:N]*wi)]';
    s     = [0 sin([1:N]*wi)]';
    c1    = [cos([1:K]*wi)]';
    s1    = [sin([1:K]*wi)]';
    Aw(:,i)= (a' * c) - j*(a'* s);
    Bw(:,i)= 1 + (b' * c1) - j*(b' * s1);
    Hew(:,i)= Aw(:,i)/Bw(:,i);
    Y =20*log10(abs(Hew));
    An=angle(Hew);
    X = w;
end
% magnitude plot
figure(1)
subplot(2,1,1)
plot(X,Y, ...
[0 wpass],[1 1],'r--', ...
[0 wpass],[-1 -1],'r--', ...
[0 wpass],[-0.0179 -0.0179],'b--', ...
[wstop pi],[-40 -40],'r--',...
[wstop pi],[-43.4090 -43.4090],'b--')
xlabel('w')
ylabel('mag H(w) in dB')
axis([0 pi -50 5]);
grid

% phase plot
subplot(2,1,2)
plot(X,An)
axis([0,pi,-10,10])
xlabel('w'), ylabel('phase H(w)')
grid

%%%%%%%%%%%%%% the results
fprintf('coefficients vector of nominator for IIR filter are: ')
a
fprintf('coefficients vector of dinominator for IIR filter are: ')
b
fprintf('the maximum passband error in db at wpass is')
    wi    = wpass;
    c     = [1 cos([1:N]*wi)]';
```

```
    s      = [0 sin([1:N]*wi)]';
    c1     = [cos([1:K]*wi)]';
    s1     = [sin([1:K]*wi)]';
    Aw     = (a' * c) - j*(a'* s);
    Bw     = 1 + (b' * c1) - j*(b' * s1);
    Hew    = Aw/Bw;
    Y      = 20*log10(abs(Hew))
fprintf('the minimum stopband attenuation in db at wstop is')
    wi     = wstop;
    c      = [1 cos([1:N]*wi)]';
    s      = [0 sin([1:N]*wi)]';
    c1     = [cos([1:K]*wi)]';
    s1     = [sin([1:K]*wi)]';
    Aw     = (a' * c) - j*(a'* s);
    Bw     = 1 + (b' * c1) - j*(b' * s1);
    Hew    = Aw/Bw;
    Y      = 20*log10(abs(Hew))

%%%%%%%%%%%%%% the transfer fuction of filter
a=y(2:5);
b=[1;y(6:8)];
TF = tf(a',b',1.25e-08,'variable','z^-1')
%%%%%%%%%%%%%% the zero-pole
[z,p,k] = tf2zpk(a,b)
%%%%%%%%%%%%%% convert to analog
Ts=1.25e-08;
F=tf(a',b',Ts);
Fc=d2c(F,'tustin')
%%%%%%%%%%%%%% The END %%%%%%%%%%%%%%%%
```

8.2 FIR_by_LMI MATLAB Code

```
% Minimax design of FIR digital filters by SDP(semidefinite programming)
% using SeDuMi(self-dual minimization)toolbox
% introduced by Fady El-Batta
clear;
clc;
N = 18;              % order of FIR filter
M = 100;             % N.O. equally-spaced sample frequencies
wpass = 0.006*pi;    % end of the passband
wstop = 0.2*pi;      % start of the stopband
t1    = 1.1e-2;      % weighting fuction W(w) value in passband
t2    = 3e-1;        % weighting fuction W(w) value in stopband
delta = 0.023;       % maximum peak passband error(ripple = 0.4 dB)

%%%%%%%%%%%%%%%%%%% Initial Values %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
wn = (wpass+wstop)/(2*pi);
K = fir1(N,wn);      % Hamming-window based
b = K';              % coefficients vector of FIR filter
x = [delta;b];       % initial value of x
Dk = [];

%%%%%%%%%%%%%%%%%% Discretization %%%%%%%%%%%%%%%%%%%%%%%%%%
w1 = linspace(0,wpass,M/2);
w2 = linspace(wstop,pi,M/2);
w  =[w1 w2];         % omega

%%%%%%%%%%%%%%%%%%% SDP problem %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% formulate the SDP problem

  for i = 1:1:M
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr    = cos(0*wi);% real part of desired frequency response
         Hi    = sin(0*wi);% imaginary part of desired frequency response
         Hd    = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr    = 0;% real part of desired frequency response
         Hi    = 0;% imaginary part of desired frequency response
         Hd    = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if     (0 <= wi) && (wi <= wpass)
                W    = t1;     % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W    = t2;     % weighting fuction W(w)
             else
                W    = 0;     % weighting fuction W(w)
             end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw   = W*Hr;
    c     = [1 cos([1:N]*wi)]';
```

```
    cw     = W*c;
    a1     = b'*cw-Hrw;
    s      = [0 sin([1:N]*wi)]';
    sw     = W*s;
    Hiw    = W*Hi;
    a2     = b'*sw+Hiw;
    Dw = [delta a1 a2;a1 1 0;a2 0 1];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  F = Dk;

 %%%%%%%%%%%%%%%%%% Using SeDuMi %%%%%%%%%%%%%%%%%%%%%%%%%
%%%%F = A0+y(1)A1+y(2)A2+......+y(19)A19+y(20)A20 >= 0
%%%%%%%%% calculate A0
Dk = [];
  for i = 1:1:M
      wi     = w(1,i);
         if      (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if      (0 <= wi) && (wi <= wpass)
                W     = t1;     % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W     = t2;     % weighting fuction W(w)
             else
                W     = 0;     % weighting fuction W(w)
             end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw    = W*Hr;
    c      = [1 cos([1:N]*wi)]';
    cw     = W*c;
    a1     = -Hrw;
    s      = [0 sin([1:N]*wi)]';
    sw     = W*s;
    Hiw    = W*Hi;
    a2     = Hiw;
    Dw = [0 a1 a2;a1 1 0;a2 0 1];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A0 = Dk;
 %%%%%%% calculate A1
s  = size(F);
```

```
r  = s(1,1);
A1 = zeros(r);
g=1;
for i=1:1:M
    A1(g,g)= 1;
    g = g+3;
end
%%%%% calculate A2
Dk = [];
  for i = 1:1:M
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr    = cos(0*wi);% real part of desired frequency response
         Hi    = sin(0*wi);% imaginary part of desired frequency response
         Hd    = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr    = 0;% real part of desired frequency response
         Hi    = 0;% imaginary part of desired frequency response
         Hd    = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if     (0 <= wi) && (wi <= wpass)
                W    = t1;    % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W    = t2;    % weighting fuction W(w)
             else
                W    = 0;    % weighting fuction W(w)
             end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw   = W*Hr;
    c     = [1 cos([1:N]*wi)]';
    cw    = W*c;
    a1    = cw(1);
    s     = [0 sin([1:N]*wi)]';
    sw    = W*s;
    Hiw   = W*Hi;
    a2    = sw(1);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A2=Dk;
  %%%%%%% calculate A3
Dk = [];
  for i = 1:1:M
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr    = cos(0*wi);% real part of desired frequency response
         Hi    = sin(0*wi);% imaginary part of desired frequency response
         Hd    = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr    = 0;% real part of desired frequency response
         Hi    = 0;% imaginary part of desired frequency response
```

```
        Hd     = complex(Hr,Hi);% desired frequency response
        end
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            if      (0 <= wi) && (wi <= wpass)
               W     = t1;     % weighting fuction W(w)
            elseif  (wstop <= wi) && (wi <= pi)
               W     = t2;     % weighting fuction W(w)
            else
               W     = 0;     % weighting fuction W(w)
            end
           %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

   Hrw    = W*Hr;
   c      = [1 cos([1:N]*wi)]';
   cw     = W*c;
   a1     = cw(2);
   s      = [0 sin([1:N]*wi)]';
   sw     = W*s;
   Hiw    = W*Hi;
   a2     = sw(2);
   Dw = [0 a1 a2;a1 0 0;a2 0 0];
   Dx = [Dw zeros(3,3*(M-1))];
   Dx = circshift(Dx,[0 3*(i-1)]);
   Dk = [Dk; Dx];

  end
  A3=Dk;
  %%%%%% calculate A4
Dk = [];
  for i = 1:1:M
      wi     = w(1,i);
        if      (0 <= wi) && (wi <= wpass)
        Hr     = cos(0*wi);% real part of desired frequency response
        Hi     = sin(0*wi);% imaginary part of desired frequency response
        Hd     = complex(Hr,Hi);% desired frequency response
        elseif (wstop <= wi) && (wi <= pi)
        Hr     = 0;% real part of desired frequency response
        Hi     = 0;% imaginary part of desired frequency response
        Hd     = complex(Hr,Hi);% desired frequency response
        end
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            if      (0 <= wi) && (wi <= wpass)
               W     = t1;     % weighting fuction W(w)
            elseif  (wstop <= wi) && (wi <= pi)
               W     = t2;     % weighting fuction W(w)
            else
               W     = 0;     % weighting fuction W(w)
            end
           %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

   Hrw    = W*Hr;
   c      = [1 cos([1:N]*wi)]';
   cw     = W*c;
   a1     = cw(3);
   s      = [0 sin([1:N]*wi)]';
   sw     = W*s;
```

```
    Hiw    = W*Hi;
    a2     = sw(3);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
    A4=Dk;
  %%%%%%%% calculate A5
Dk = [];
  for i = 1:1:M
      wi     = w(1,i);
         if      (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if      (0 <= wi) && (wi <= wpass)
                W     = t1;     % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W     = t2;     % weighting fuction W(w)
             else
                W     = 0;     % weighting fuction W(w)
             end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw    = W*Hr;
    c      = [1 cos([1:N]*wi)]';
    cw     = W*c;
    a1     = cw(4);
    s      = [0 sin([1:N]*wi)]';
    sw     = W*s;
    Hiw    = W*Hi;
    a2     = sw(4);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A5=Dk;
 %%%%%%%% calculate A6
Dk = [];
  for i = 1:1:M
      wi     = w(1,i);
         if      (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
```

```
        Hr    = 0;% real part of desired frequency response
        Hi    = 0;% imaginary part of desired frequency response
        Hd    = complex(Hr,Hi);% desired frequency response
        end
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            if      (0 <= wi) && (wi <= wpass)
               W     = t1;     % weighting fuction W(w)
            elseif  (wstop <= wi) && (wi <= pi)
               W     = t2;     % weighting fuction W(w)
            else
               W     = 0;     % weighting fuction W(w)
            end
           %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw   = W*Hr;
    c     = [1 cos([1:N]*wi)]';
    cw    = W*c;
    a1    = cw(5);
    s     = [0 sin([1:N]*wi)]';
    sw    = W*s;
    Hiw   = W*Hi;
    a2    = sw(5);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A6=Dk;
  %%%%%%% calculate A7
Dk = [];
  for i = 1:1:M
      wi    = w(1,i);
        if      (0 <= wi) && (wi <= wpass)
        Hr    = cos(0*wi);% real part of desired frequency response
        Hi    = sin(0*wi);% imaginary part of desired frequency response
        Hd    = complex(Hr,Hi);% desired frequency response
        elseif (wstop <= wi) && (wi <= pi)
        Hr    = 0;% real part of desired frequency response
        Hi    = 0;% imaginary part of desired frequency response
        Hd    = complex(Hr,Hi);% desired frequency response
        end
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            if      (0 <= wi) && (wi <= wpass)
               W     = t1;     % weighting fuction W(w)
            elseif  (wstop <= wi) && (wi <= pi)
               W     = t2;     % weighting fuction W(w)
            else
               W     = 0;     % weighting fuction W(w)
            end
           %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw   = W*Hr;
    c     = [1 cos([1:N]*wi)]';
    cw    = W*c;
    a1    = cw(6);
```

```
    s      = [0 sin([1:N]*wi)]';
    sw     = W*s;
    Hiw    = W*Hi;
    a2     = sw(6);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A7=Dk;
  %%%%%% calculate A8
Dk = [];
  for i = 1:1:M
      wi     = w(1,i);
         if      (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if      (0 <= wi) && (wi <= wpass)
                W     = t1;     % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W     = t2;     % weighting fuction W(w)
             else
                W     = 0;     % weighting fuction W(w)
             end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw    = W*Hr;
    c      = [1 cos([1:N]*wi)]';
    cw     = W*c;
    a1     = cw(7);
    s      = [0 sin([1:N]*wi)]';
    sw     = W*s;
    Hiw    = W*Hi;
    a2     = sw(7);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A8=Dk;
 %%%%%%% calculate A9
Dk = [];
  for i = 1:1:M
      wi     = w(1,i);
         if      (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
```

```
        Hd     = complex(Hr,Hi);% desired frequency response
        elseif (wstop <= wi) && (wi <= pi)
        Hr     = 0;% real part of desired frequency response
        Hi     = 0;% imaginary part of desired frequency response
        Hd     = complex(Hr,Hi);% desired frequency response
        end
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            if      (0 <= wi) && (wi <= wpass)
               W     = t1;     % weighting fuction W(w)
            elseif  (wstop <= wi) && (wi <= pi)
               W     = t2;     % weighting fuction W(w)
            else
               W     = 0;     % weighting fuction W(w)
            end
           %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

   Hrw    = W*Hr;
   c      = [1 cos([1:N]*wi)]';
   cw     = W*c;
   a1     = cw(8);
   s      = [0 sin([1:N]*wi)]';
   sw     = W*s;
   Hiw    = W*Hi;
   a2     = sw(8);
   Dw = [0 a1 a2;a1 0 0;a2 0 0];
   Dx = [Dw zeros(3,3*(M-1))];
   Dx = circshift(Dx,[0 3*(i-1)]);
   Dk = [Dk; Dx];

  end
  A9=Dk;
  %%%%% calculate A10
Dk = [];
  for i = 1:1:M
      wi     = w(1,i);
        if      (0 <= wi) && (wi <= wpass)
        Hr     = cos(0*wi);% real part of desired frequency response
        Hi     = sin(0*wi);% imaginary part of desired frequency response
        Hd     = complex(Hr,Hi);% desired frequency response
        elseif (wstop <= wi) && (wi <= pi)
        Hr     = 0;% real part of desired frequency response
        Hi     = 0;% imaginary part of desired frequency response
        Hd     = complex(Hr,Hi);% desired frequency response
        end
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            if      (0 <= wi) && (wi <= wpass)
               W     = t1;     % weighting fuction W(w)
            elseif  (wstop <= wi) && (wi <= pi)
               W     = t2;     % weighting fuction W(w)
            else
               W     = 0;     % weighting fuction W(w)
            end
           %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

   Hrw    = W*Hr;
   c      = [1 cos([1:N]*wi)]';
```

```
    cw     = W*c;
    a1     = cw(9);
    s      = [0 sin([1:N]*wi)]';
    sw     = W*s;
    Hiw    = W*Hi;
    a2     = sw(9);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A10=Dk;
%%%%%% calculate A11
Dk = [];
  for i = 1:1:M
      wi     = w(1,i);
         if      (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if      (0 <= wi) && (wi <= wpass)
                W     = t1;     % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W     = t2;     % weighting fuction W(w)
             else
                W     = 0;     % weighting fuction W(w)
             end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw    = W*Hr;
    c      = [1 cos([1:N]*wi)]';
    cw     = W*c;
    a1     = cw(10);
    s      = [0 sin([1:N]*wi)]';
    sw     = W*s;
    Hiw    = W*Hi;
    a2     = sw(10);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A11=Dk;
%%%%%% calculate A12
Dk = [];
  for i = 1:1:M
      wi     = w(1,i);
         if      (0 <= wi) && (wi <= wpass)
```

```
        Hr    = cos(0*wi);% real part of desired frequency response
        Hi    = sin(0*wi);% imaginary part of desired frequency response
        Hd    = complex(Hr,Hi);% desired frequency response
        elseif (wstop <= wi) && (wi <= pi)
        Hr    = 0;% real part of desired frequency response
        Hi    = 0;% imaginary part of desired frequency response
        Hd    = complex(Hr,Hi);% desired frequency response
        end
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            if      (0 <= wi) && (wi <= wpass)
               W    = t1;     % weighting fuction W(w)
            elseif  (wstop <= wi) && (wi <= pi)
               W    = t2;     % weighting fuction W(w)
            else
               W    = 0;     % weighting fuction W(w)
            end
           %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

   Hrw   = W*Hr;
   c     = [1 cos([1:N]*wi)]';
   cw    = W*c;
   a1    = cw(11);
   s     = [0 sin([1:N]*wi)]';
   sw    = W*s;
   Hiw   = W*Hi;
   a2    = sw(11);
   Dw = [0 a1 a2;a1 0 0;a2 0 0];
   Dx = [Dw zeros(3,3*(M-1))];
   Dx = circshift(Dx,[0 3*(i-1)]);
   Dk = [Dk; Dx];

  end
  A12=Dk;
  %%%%%%% calculate A13
Dk = [];
  for i = 1:1:M
      wi    = w(1,i);
        if      (0 <= wi) && (wi <= wpass)
        Hr    = cos(0*wi);% real part of desired frequency response
        Hi    = sin(0*wi);% imaginary part of desired frequency response
        Hd    = complex(Hr,Hi);% desired frequency response
        elseif (wstop <= wi) && (wi <= pi)
        Hr    = 0;% real part of desired frequency response
        Hi    = 0;% imaginary part of desired frequency response
        Hd    = complex(Hr,Hi);% desired frequency response
        end
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            if      (0 <= wi) && (wi <= wpass)
               W    = t1;     % weighting fuction W(w)
            elseif  (wstop <= wi) && (wi <= pi)
               W    = t2;     % weighting fuction W(w)
            else
               W    = 0;     % weighting fuction W(w)
            end
           %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
    Hrw   = W*Hr;
    c     = [1 cos([1:N]*wi)]';
    cw    = W*c;
    a1    = cw(12);
    s     = [0 sin([1:N]*wi)]';
    sw    = W*s;
    Hiw   = W*Hi;
    a2    = sw(12);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A13=Dk;
%%%%% calculate A14
Dk = [];
  for i = 1:1:M
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if     (0 <= wi) && (wi <= wpass)
                W    = t1;    % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W    = t2;    % weighting fuction W(w)
             else
                W    = 0;    % weighting fuction W(w)
             end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw   = W*Hr;
    c     = [1 cos([1:N]*wi)]';
    cw    = W*c;
    a1    = cw(13);
    s     = [0 sin([1:N]*wi)]';
    sw    = W*s;
    Hiw   = W*Hi;
    a2    = sw(13);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A14=Dk;
 %%%%% calculate A15
Dk = [];
  for i = 1:1:M
```

```
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if     (0 <= wi) && (wi <= wpass)
                W    = t1;     % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W    = t2;     % weighting fuction W(w)
             else
                W    = 0;     % weighting fuction W(w)
             end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw   = W*Hr;
    c     = [1 cos([1:N]*wi)]';
    cw    = W*c;
    a1    = cw(14);
    s     = [0 sin([1:N]*wi)]';
    sw    = W*s;
    Hiw   = W*Hi;
    a2    = sw(14);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A15=Dk;
%%%%%% calculate A16
Dk = [];
  for i = 1:1:M
      wi    = w(1,i);
         if     (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if     (0 <= wi) && (wi <= wpass)
                W    = t1;     % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W    = t2;     % weighting fuction W(w)
             else
                W    = 0;     % weighting fuction W(w)
             end
```

```
             %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw    = W*Hr;
    c      = [1 cos([1:N]*wi)]';
    cw     = W*c;
    a1     = cw(15);
    s      = [0 sin([1:N]*wi)]';
    sw     = W*s;
    Hiw    = W*Hi;
    a2     = sw(15);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A16=Dk;
  %%%%%% calculate A17
Dk = [];
  for i = 1:1:M
      wi    = w(1,i);
         if      (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if      (0 <= wi)  && (wi <= wpass)
                W     = t1;     % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W     = t2;     % weighting fuction W(w)
             else
                W     = 0;     % weighting fuction W(w)
             end
             %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw    = W*Hr;
    c      = [1 cos([1:N]*wi)]';
    cw     = W*c;
    a1     = cw(16);
    s      = [0 sin([1:N]*wi)]';
    sw     = W*s;
    Hiw    = W*Hi;
    a2     = sw(16);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A17=Dk;
%%% calculate A18
```

```
Dk = [];
  for i = 1:1:M
      wi    = w(1,i);
         if      (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if      (0 <= wi) && (wi <= wpass)
                W     = t1;     % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W     = t2;     % weighting fuction W(w)
             else
                W     = 0;     % weighting fuction W(w)
             end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw    = W*Hr;
    c      = [1 cos([1:N]*wi)]';
    cw     = W*c;
    a1     = cw(17);
    s      = [0 sin([1:N]*wi)]';
    sw     = W*s;
    Hiw    = W*Hi;
    a2     = sw(17);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A18=Dk;
%%%% calculate A19
Dk = [];
  for i = 1:1:M
      wi    = w(1,i);
         if      (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if      (0 <= wi) && (wi <= wpass)
                W     = t1;     % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W     = t2;     % weighting fuction W(w)
             else
```

```
                W     = 0;     % weighting fuction W(w)
             end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw     = W*Hr;
    c       = [1 cos([1:N]*wi)]';
    cw      = W*c;
    a1      = cw(18);
    s       = [0 sin([1:N]*wi)]';
    sw      = W*s;
    Hiw     = W*Hi;
    a2      = sw(18);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
  A19=Dk;
 %%% calculate A20
Dk = [];
  for i = 1:1:M
      wi     = w(1,i);
         if      (0 <= wi) && (wi <= wpass)
         Hr     = cos(0*wi);% real part of desired frequency response
         Hi     = sin(0*wi);% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         elseif (wstop <= wi) && (wi <= pi)
         Hr     = 0;% real part of desired frequency response
         Hi     = 0;% imaginary part of desired frequency response
         Hd     = complex(Hr,Hi);% desired frequency response
         end
         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
             if      (0 <= wi) && (wi <= wpass)
                W     = t1;     % weighting fuction W(w)
             elseif  (wstop <= wi) && (wi <= pi)
                W     = t2;     % weighting fuction W(w)
             else
                W     = 0;     % weighting fuction W(w)
             end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    Hrw     = W*Hr;
    c       = [1 cos([1:N]*wi)]';
    cw      = W*c;
    a1      = cw(19);
    s       = [0 sin([1:N]*wi)]';
    sw      = W*s;
    Hiw     = W*Hi;
    a2      = sw(19);
    Dw = [0 a1 a2;a1 0 0;a2 0 0];
    Dx = [Dw zeros(3,3*(M-1))];
    Dx = circshift(Dx,[0 3*(i-1)]);
    Dk = [Dk; Dx];

  end
```

```
  A20=Dk;
  %%%%%%%%%%%%%%%%SeDuMi function
 p = length([1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]);
bt = -[1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];
ct = vec(A0);
K.s = size(A0,1);
At(:,1) = -vec(A1);
At(:,2) = -vec(A2);
At(:,3) = -vec(A3);
At(:,4) = -vec(A4);
At(:,5) = -vec(A5);
At(:,6) = -vec(A6);
At(:,7) = -vec(A7);
At(:,8) = -vec(A8);
At(:,9) = -vec(A9);
At(:,10) = -vec(A10);
At(:,11) = -vec(A11);
At(:,12) = -vec(A12);
At(:,13) = -vec(A13);
At(:,14) = -vec(A14);
At(:,15) = -vec(A15);
At(:,16) = -vec(A16);
At(:,17) = -vec(A17);
At(:,18) = -vec(A18);
At(:,19) = -vec(A19);
At(:,20) = -vec(A20);
[x, y, info] = sedumi(At,bt,ct,K);
y
fa=y(2:20);
%%%%%%%%%%%%%%%%%%%%%%% Plot %%%%%%%%%%%%%%%%%%%%%%%%%%%
b=y(2:20);
N =18;                  % order of FIR filter
M =100;                 % the number of sample frequencies
j = sqrt(-1);
wpass = 0.006*pi;       % end of the passband
wstop = 0.2*pi;         % start of the stopband
w  = linspace(0,pi,M); % omega

for i = 1:1:M
    wi    = w(1,i);
    c     = [1 cos([1:N]*wi)]';
    s     = [0 sin([1:N]*wi)]';
    Hew(:,i)= (b' * c) - j*(b'* s);
    Y =20*log10(abs(Hew));
    An=angle(Hew);
    X = w;
end
% magnitude plot
figure(1)
subplot(2,1,1)
plot(X,Y, ...
[0 wpass],[1 1],'r--', ...
[0 wpass],[-1 -1],'r--', ...
[0 wpass],[-0.3167 -0.3167],'b--', ...
[wstop pi],[-40 -40],'r--',...
[wstop pi],[-44.2459 -44.2459],'b--')
```

```
xlabel('w')
ylabel('mag H(w) in dB')
axis([0 pi -50 5]);
grid

% phase plot
subplot(2,1,2)
plot(X,An)
axis([0,pi,-10,10])
xlabel('w'), ylabel('phase H(w)')
grid

%%%%%%%%%%%%%% the results
fprintf('coefficients vector of FIR filter are: ')
b
fprintf('the maximum passband error in db at wpass is')
    wi    = wpass;
    c     = [1 cos([1:N]*wi)]';
    s     = [0 sin([1:N]*wi)]';
    Hew   = (b' * c) - j*(b'* s);
    Y     = 20*log10(abs(Hew))
fprintf('the minimum stopband attenuation in db at wstop is')
    wi    = wstop;
    c     = [1 cos([1:N]*wi)]';
    s     = [0 sin([1:N]*wi)]';
    Hew   = (b' * c) - j*(b'* s);
    Y     = 20*log10(abs(Hew))
 %%%%%%%%%%%%%% The END %%%%%%%%%%%%%%%%%
```

8.3 FIR_ min_order MATLAB Code

```
% Minimize order of a lowpass FIR filter %(magnitude design)
% by Fady El-Batta
%
% Designs an FIR low pass filter using spectral factorization method where
%we:
% - minimize the filter order
% - have a constraint on the maximum passband ripple
% - have a constraint on the maximum stopband attenuation
%
%   minimize   filter order n
%       s.t.   1/delta <= |H(w)| <= delta   for w in the passband
%              |H(w)| <= atten_level        for w in the stopband
%
% We change variables via spectral factorization method and get:
%
%   minimize   filter order n
%       s.t.   (1/delta)^2 <= R(w) <= delta^2  for w in the passband
%              R(w) <= atten_level^2           for w in the stopband
%              R(w) >= 0                       for all w
%
% where R(w) is the squared magnited of the frequency response
% (and the Fourier transform of the autocorrelation coefficients r).
% Variables are coeffients r. delta is the allowed passband ripple
% and atten_level is the max allowed level in the stopband.
%
% This is a quasiconvex problem and can be solved using a bisection.
%
%*********************************************************************
% user's filter specs (for a low-pass filter example)
%*********************************************************************
% filter order that is used to start the bisection (has to be feasible)
max_order = 20;        % FIR filter length = FIR filter order + 1 = 19 +1=20
wpass = 0.006*pi;        % passband cutoff freq (in radians)
wstop = 0.2*pi;          % stopband start freq (in radians)
delta = 0.3167;          % max (+/-) passband ripple in dB
atten_level = -44.2459;   % stopband attenuation level in dB

%********************************************************************
% create optimization parameters
%********************************************************************
m = 100;
w = linspace(0,pi,m);

%*********************************************************************
% use bisection algorithm to solve the problem
%*********************************************************************
cvx_quiet(true);

n_bot = 1;
n_top = max_order;

while( n_top - n_bot > 1)
```

```
  % try to find a feasible design for given specs
  n_cur = ceil( (n_top + n_bot)/2 );

  % create optimization matrices
  % A is the matrix used to compute the power spectrum
  % A(w,:) = [1 2*cos(w) 2*cos(2*w) ... 2*cos(n*w)]
  A = [ones(m,1) 2*cos(kron(w',[1:n_cur-1]))];

  % passband 0 <= w <= w_pass
  ind = find((0 <= w) & (w <= wpass));    % passband
  Ap  = A(ind,:);

  % transition band is not constrained (w_pass <= w <= w_stop)

  % stopband (w_stop <= w)
  ind = find((wstop <= w) & (w <= pi));   % stopband
  As  = A(ind,:);

  % formulate and solve the feasibility linear-phase lp filter design
  cvx_begin
    variable r(n_cur,1);
    % feasibility problem
    % passband bounds
    Ap*r <= (10^(delta/20))^2;
    Ap*r >= (10^(-delta/20))^2;
    % stopband bounds
    abs( As*r ) <= (10^(atten_level/20))^2;
    % nonnegative-real constraint for all frequencies (a bit redundant)
    A*r >= 0;
  cvx_end

  % bisection
  if strfind(cvx_status,'Solved') % feasible
    fprintf(1,'Problem is feasible for filter order = %d taps\n',n_cur);
    n_top = n_cur;
    % construct the original impulse response
    h = spectral_fact(r);
  else % not feasible
    fprintf(1,'Problem not feasible for filter order = %d taps\n',n_cur);
    n_bot = n_cur;
  end
end

n = n_top;
fprintf(1,'\nOptimum number of filter taps for given specs is %d.\n',n);

cvx_quiet(false);

%*******************************************************************
% plots
%*******************************************************************
figure(1)
% FIR impulse response
plot([0:n-1],h','o',[0:n-1],h','b:')
xlabel('t'), ylabel('h(t)')
figure(2)
```

```
% frequency response
H = exp(-j*kron(w',[0:n-1]))*h;
% magnitude
subplot(2,1,1)
plot(w,20*log10(abs(H)),...
     [wstop pi],[atten_level atten_level],'r--',...
     [0 wpass],[delta delta],'r--',...
     [0 wpass],[-delta -delta],'r--');
axis([0,pi,-40,10])
xlabel('w'), ylabel('mag H(w) in dB')
% phase
subplot(2,1,2)
plot(w,angle(H))
axis([0,pi,-pi,pi])
```

8.4 List of Acronyms

Acronyms	Meaning
CPE	Customer Premises Equipment
CSMA	Carrier Sense Multiple Access
CVX	Convex software
DSL	Digital Subscriber Line
FDD	Frequency Division Duplexing
FIR	Finite Impulse Response
GPS	Global Positioning System
IIR	Infinite Impulse Response
LF	Loop Filter
LMI	Linear Matrix Inequality
LOS	Line of Sight
LP	Linear Programming
MAC	Media Access Control
MIMO	Multi-Input Multi-Output
NLOS	Non Line of Sight
OFDMA	Orthogonal Frequency Division Multiple Access
OFDM	Orthogonal Frequency Division Multiplexing
PFD	Phase/Frequency Detector
PLL	Phase Locked Loop
QoS	Quality of Service
SDP	Semi-Definite Programming
TDD	Time Division Duplexing

SeDuMi	Self-Dual Minimization
VCO	Voltage Controlled Oscillator
WiFi	Wireless Fidelity
WiMAX	Worldwide Interoperability for Microwave Access
WMAN	Wireless Metropolitan Area Network
XOR	Exclusive OR