Ref: SUST/ CGS/A11

Approval Page

(To be completed after the college council approval)

Name of Candidate: Mohammed | Al-Hussien | Mohammed | Ahmed

Thesis title: Implementation of Digital FM Receiver using FPGA

Degree Examined for: M.Sc

Approved by:

1. External Examiner

Name: Dr. Amin Babiker A/Mallei Mustafa

Signature: ..... Date: 6/6/2018

2. Internal Examiner

Name: Dr. Mudathir Abdallah Osman Fagiri

Signature: ..... Date: 6-6-2018

3. Supervisor

Name: Dr. Rashid A/Halem Saeed

Signature: ..... Date: 06/06/2018

Sudan University of Science and Technology
College of Graduate Studies
M.Sc. Program in Communication Engineering

Implementation of Digital FM Receiver using FPGA

تنفيذ مستقبل رقمي لإشارة تضمين التردد باستخدام مصفوفة البوابات المنطقية القابلة للبرمجة

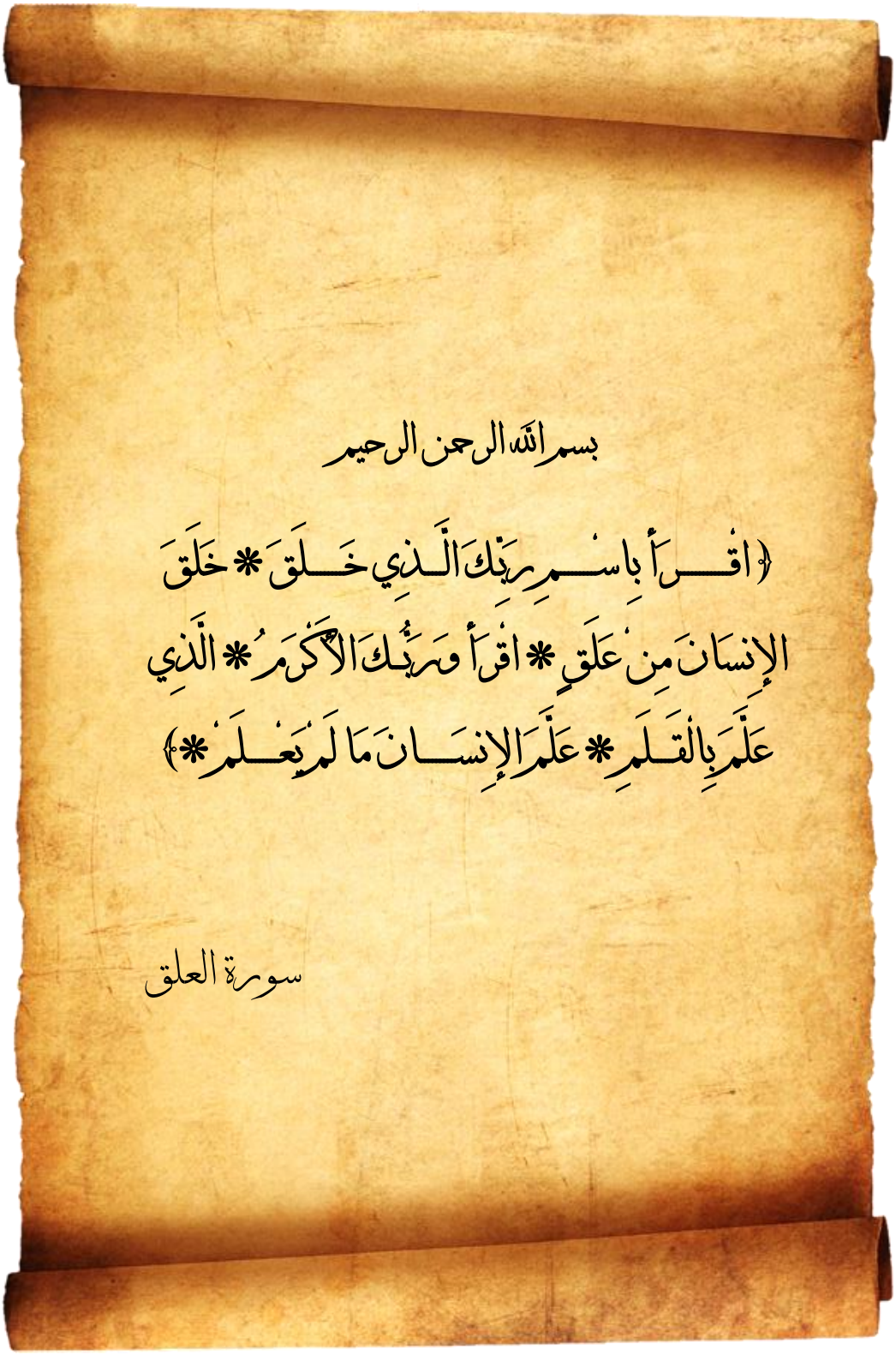
A Thesis Submitted to Sudan University of Science and Technology in Partial Fulfillment of the Requirements for the Degree of M.Sc. in Communication Engineering

Prepared by
MOHAMMAD AL-HUSIEN MOHAMMAD AHMED MAKKAWI

Supervisor
Prof. Dr. RASHID A. SAEED

June, 2018

الآية


بسم الله الرحمن الرحيم
﴿ اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ * خَلَقَ
الإِنسَانَ مِنْ عَلَقٍ * اقْرَأْ وَرَبُّكَ الأَكْرَمُ * الَّذِي
عَلَّمَ بِالْقَلَمِ * عَلَّمَ الإِنسَانَ مَا لَمْ يَعْلَمْ * ﴾
سورة العلق

DEDICATION

To the spirit of my late father, God rest his soul

Acknowledgements

Blessings and peace be upon the most noble of Messengers, the Prophet Mohammad and on his kinsman and disciples.

Firstly, I would like to thank my parents for encouraging me at all my educational levels. My thanks and appreciation go too to Eng. Marwa Alhaj for her unlimited support in this thesis. Also, all thanks to my supervisor Prof. Dr. RASHID A. SAEED for his continuous support in this thesis until reaching this state. My thanks also go to my coworker Eng. Montaser Abdel Jalil and all staff of telecommunication research center (TRC) for the great support and cooperation. Last but not least, A special thanks go to all my friends.

Abstract

The complexity of hardware in traditional FM radio lead to bigger radio size, noisy, difficulties in maintenance and higher production costs. In addition to limitation on protocols and frequency bands supported, which lead to use different radio devices for different networks and communication protocols. Software Defined Radio technology provides an efficient and comparatively inexpensive solution to these problems replacing a lot of cascaded electronics components by a single digital chip FPGA (Field Programmable Gate Array) resulting in low production cost, immunity to noise, ease of maintenance and enormous reduction of radio size. In this thesis, digital FM receiver was designed in SIMULINK and implemented successfully on Xilinx Zynq-7000 FPGA on ZedBoard evaluation kit with JTAG co-simulation. The design was tested with both single tone message and multi tone message real voice. Receiver was tunable in real time to select receiving FM channel with GUI designed for this project. Results achieved were very acceptable and transmitted message was successfully recovered on the receiver which was clear from time domain graphs from time scope and frequency domain graphs from spectrum analyzer.

المستخلص

يؤدي التعقيد في التكوينات لأجهزة راديو تضمين التردد التقليدية إلى كبر حجم الراديو ، وتعرضه للضجيج، وصعوبات في صيانته وارتفاع تكلفة إنتاجه. بالإضافة إلى المحدودية في البرتوكولات ونطاقات التردد المدعومة مما يؤدي إلى استخدام أجهزة راديو متعددة لمختلف الشبكات وبروتوكولات الاتصال. توفر تقنية الراديو المعرف برمجيا حلًا فعالاً وغير مكلف نسبيًا لهذه المشكلات حيث تستبدل الكثير من المكونات الإلكترونية برقاقة رقمية واحدة (مصفوفة البوابات المنطقية القابلة للبرمجة) مما يؤدى إلى انخفاض تكلفة إنتاج الراديو ومناعته ضد الضجيج و سهولة صيانته بالإضافة إلى الخفض الهائل في حجم الراديو. في هذه الأطروحة ، تم تصميم جهاز راديو تضمين التردد الرقمي باستخدام بيئة سيميولينك ونُفذ بنجاح على رقاقة زايلينيكس زينك 7000 للبوابة المنطقية القابلة للبرمجة على لوحة التقييم زد بورد بواسطة المحاكاة المشتركة عبر كابل الجيتاج. تم اختبار التصميم من خلال رسالة صوتية أحادية التردد وأخرى بصوت حقيقي متعدد التردد. أمكن ضبط المستقبل في الوقت الفعلي لاختيار تلقي موجات تضمين التردد باستخدام واجهة المستخدم الرسومية المصممة لهذا المشروع. كانت النتائج المحققة مقبولة للغاية وتم بنجاح استرجاع الرسالة المرسلة على المستقبِل والتي كانت واضحة من خلال المخططات البيانية في النطاق الزمني و المخططات البيانية في النطاق الترددي من محلل الأطياف.

.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ADC	Analog to Digital Converter
AGC	Automatic Gain Control
AM	Amplitude Modulation
ASIC	Application Specific Integrated Circuit
BPF	Band Pass Filter
CB	Citizens Band
CDMA	Code Division Multiple Access
DAC	Digital to Analog Converter
DARPA	Defense Advanced Research Projects Agency
DDS	Direct Digital Synthesizer
DSP	Digital Signal Processor
FIR	Finite duration Impulse Response
FM	Frequency Modulation
FPGA	Field Programmable Gate Array
GPP	General Purpose Preprocessor

GUI	Graphical User Interface
I	In-phase component
IEEE	Institute of Electrical and Electronics Engineers
IF	Intermediate Frequency
IIR	Infinite duration Impulse Response
IP core	Intellectual Property core
ITU	International Telecommunication Union
JTRS	Joint Tactical Radio System
LNA	Low Noise Amplifier
LO	Local Oscillator
LPF	Low Pass Filter
LTE	Long Term Evolution
PA	Power Amplifier
Q	Quadrature component
R&D	Research and Development
RF	Radio Frequency
SDR	Software Defined Radio
WIFI	Wireless Fidelity, IEEE 802.11b wireless networking

Chapter One

INTRODUCTION

Chapter One

1 Introduction

1.1 Background

Traditionally, communication systems required specialized hardware to implement their functionality. In order to keep production costs low, these systems contained only the hardware necessary to perform the tasks that they were designed for. As a result, these systems were often difficult to modify and upgrade. Today, digital signal processor (DSP) technology has evolved to the point where many of the encoders, modulators, filters and decoders used by these communication systems can be implemented in software [1]. Some systems rely on software to perform small tasks while others implement all of their baseband functionality in software. Those systems that implement all of their baseband functionality in software are called software defined radios (SDR) [2].

SDR are becoming a desirable alternative to traditional radio systems due, in large part, to their flexibility. SDRs have the ability to be reprogrammed, which means that they are upgradeable to a certain extent. Future-proof may be too strong a promise but the point is an appropriately made. SDR can support a very large set of waveforms and coding schemes. This is a very attractive option because as technologies advance and waveforms change, traditional systems require costly replacement while SDRs may require simple upgrades. SDRs can also, in some cases, replace a set of many traditional radios [3]. This can reduce operating costs as well as space requirements while maintaining or improving functionality of the original system.

Until recently, analog receivers and modulation techniques have been unsurpassed in performance. However, new technologies in digital

communications are utilized in developing high-speed modems, spread-spectrum systems, next-generation cellular radios, and many other digital systems that dwarf their analog counterparts.

1.2 Literature Review

Over the last few years, analog radio systems are being replaced by digital radio systems for various radio applications in military, civilian and commercial spaces. In addition to this, programmable hardware modules are increasingly being used in digital radio systems at different functional levels. SDR technology aims to take advantage of these programmable hardware modules to build open architecture based radio system software [4].

The idea of implementing radio functions in software rather than hardware had already been established when, in 1991, Joseph Mitola coined the term 'Software Radio' [5]. He said "*A software radio is a radio whose channel modulation waveforms are defined in software. That is, digital waveforms are generated and converted to analog form using a Digital to Analog Converter (DAC) at the transmitter and at the receiver, they are converted to digital form using Analog to Digital Converter (ADC) and demodulated using the software*" [5]. SDR Technology was originally conceived as a means to facilitate better communications between the different forces of the US military. SDR was seen as a technology that could facilitate the interworking of all different radios and radio systems used within the forces. The aim was to eventually reduce the number of different radio systems used by the different military services.

SDR Technology has been on the research agenda for more than 20 years. During this period both the original intended use, as well as the associated implications, have changed. The basic idea of SDR Technology is to allow the functionality of communication devices to be similar to that of

personal computers. When strictly following this principle, SDR Technology mainly reduces the number of communication platforms required but it does not introduce new system functionality. Hence it can be seen as an enabler rather than as complete system technology [6]. If SDR technology is properly applied, it will facilitate this single platform design, and will also provide a path towards the realization of concepts such as re-configurability, run-time reconfiguration, and eventually self-governed learning (cognitive) radio. These technologies can be very useful in enabling new application areas such as dynamic handling of spectrum and radio resources.

There are many Research and development programs and initiatives investigating possibilities for SDR system architectures and supporting technologies. Supporting technologies include antennas, performance of chip sets, battery technologies and RF digitization. Most R&D programs tackle either individual technologies, combinations of several supporting technologies or the platform architecture [6].

In the military domain research efforts include SPEAKeasy, the US Joint Tactical Radio System (JTRS) program, the DARPA next generation (XG) program or the Finish Software Radio Program. SDR research in the commercial domain ranges from international initiatives (e.g. European 5^{th} and 6^{th} framework research projects) to national programs (e.g. Réseau National de Recherche en Télécommunications) and commercial products. The report documents the approaches investigated within the most significant projects researching SDR, this aims to help understanding the capabilities and possibilities of SDR Technology.

SDR refers to the class of reprogrammable radios in which the same piece of hardware can perform different functions at different times. A convenient approach is to use Field Programmable Gate Arrays (FPGAs)

to implement physical layer signal processing for SDRs. The FPGAs was first invented by Xilinx in 1984 [7]. Since its invention it has gone from being simple glue logic chips to actually replacing custom application–specific integrated circuits (ASICs) and processors for signal processing and control applications.

Currently the radio interface in a wireless system is usually implemented by dedicated hardware. SDR came out as reprogrammable architectures were developed with the throughput required to perform the signal processing operations common in radios. Many of the SDR developments have been increased by the need for reconfigurable radio receivers and transmitters [4]. For example, the radio system at a cell site might need to be reconfigured to implement new standards or protocols, bypass failed hardware, or reconfigure the network due to a change in the traffic pattern. By using SDR, these changes can be performed remotely, and without changing the radio's hardware, thereby reducing the need for visits to the site and increasing cost efficiency [8].

Real time SDR designs can be implemented using a variety of digital hardware platforms. These include FPGAs, digital signal processors, ASICs and general purpose processors.

The DSP platform is essentially a microprocessor based system optimized for digital signal processing applications. High level languages such as C and MATLAB are used to program DSPs. One of the main disadvantages of DSPs are that when there are several computations to be performed, parallel executions of these computations will slow down the rate at which data is processed.

FPGAs are general purpose integrated circuits that are programmed by a hardware designer. An FPGA is programmed by downloading a bitstream into the static on-chip random access memory. The unique feature of

FPGA is that it can be reprogrammed, even after it has been deployed in the system. Because of this unique quality, their configuration can be easily modified to upgrade the system.

ASICs, as the name implies, are used for only specific applications. ASICs implement circuits with fixed hardware and cannot be modified after manufacture. Typically, ASIC design is very expensive and time consuming.

General purpose processors are similar to DSPs as a hardware platform. It offers flexibility and simplicity of design. Like DSPs, radio functionalities can be implemented in high level languages such as C and C++. General purpose processors are most generalized hardware that can be programmed to perform various functions, while ASICs are designed for a specific application.

FPGAs offer a compromise in flexibility between ASIC and processor based platforms. There is a tradeoff between the maximum flexibility and high power consumption of processor platforms and the minimum flexibility and lower power consumption of ASICs with FPGAs lying in between. There are other advantages of using FPGAs instead of DSPs for signal processing in telecommunication system. FPGAs are small, faster, offer higher throughputs, while the DSPs are cheaper than FPGAs.

SDR for communication systems places several stringent requirements on the Analog to Digital Converter (ADC). Software radio architecture can support multiple standards by performing analog to digital conversion of the radio frequency (RF) signals. The ADC must have a sufficiently fast sampling frequency and with SNR to accurately reconstruct the received signal. Such an ADC can be prohibitively expensive, as well as challenging to interface with a DSP or FPGA.

1.3 Problem Statement

Traditional hardware based FM radio devices can support limited protocols and frequency bands which lead to use different radio devices for different networks and communication protocols. Also, the complexity of hardware in traditional radio lead to bigger radio size, noisy, difficulties in maintenance and higher production costs.

1.4 Proposed Solution

SDR technology provides an efficient and comparatively inexpensive solution to these problems, performing the baseband signal processing in the software domain allows for much greater re-configurability and selectivity for any frequency band or communication protocol than traditional FM radios provide. Also, replacing a lot of cascaded electronics components by a single digital chip FPGA result in low production cost, immunity to noise, ease of maintenance and enormous reduction of radio size.

1.5 Aim and Objectives

The main aim of this research is to design and implementation of FM receiver based on SDR methodology using MATLAB and FPGA chip for verification. The objectives are according to the following stages:

1- To study conventional FM receiver.
2- To convert analog blocks of the FM receiver to software defined functions using SIMULINK environment in MATLAB.
3- To generate bit file using Hardware Co-Simulation with Vivado System Generator software.
4- Verifying design by download bit file in FPGA.

1.6 Methodology

In this research, the methodology intended is to design, simulate, and implement a FM SDR receiver using SIMULINK environment in MATLAB software with Xilinx Block set. An evaluation kit (ZedBoard) will be used for testing as well as implementing design on real FPGA chip (Xilinx Zynq-7000).

1.7 Thesis Outlines

In this thesis, the fundamentals of SDR are first presented in Chapter II, which includes the general background information and various definition for SDR. This includes the difference between the SDR and conventional radio, characteristics and advantages of SDR, possible design issues, and the platform choices for implementing SDR based wireless communication systems.

In Chapter III, FM receiver architecture is presented. This include a brief introduction of FM demodulation theory with explanation of each stage of the receiving process.

Chapter IV, presents the design and simulation set up for implementing FM receiver on FPGA platform, using Hardware Co-Simulation with Vivado System Generator software and Simulink. Results of the simulation and output waveforms are presented and analyzed. The implementation of the design will be made on the ZedBoard Zynq-7000 platform and results are compared with the simulation results.

Chapter V, summarizes the results and comments of the design, simulation and implementation. Also, recommendations and future works are discussed.

Chapter Two

LITERATURE REVIEW

Chapter Two

2 Literature Review

2.1 Overview

Radios are essential parts of everyday human communications, whether people realize it or not. When most people think of radios, they think of the AM/FM radios in their cars, hand-held two-way radios, or CB radios. However, radios are much more prevalent in society than most realize. For instance, the Wi-Fi adapters within a computer or smart-device are radios and Bluetooth earpieces used to talk on the phone are radios. In general, a radio is any device that transmits or receives information wirelessly using electromagnetic waves known as radio waves.

Classically, radios have been made from pieces of hardware designed for use in one specific radio. These radios can be referred to as hardware-defined radios because the radio is completely dependent on the hardware such as electrical circuits and electronic devices. However, software-defined radios, developed in the past few years, are a new type of radio in which the type of radio is determined by a piece of software. Using software rather than hardware to implement some stages of a radio system enables a radio to be more easily configured, modified, and developed for multiple systems [9].

2.2 Software Defined Radio Definition

A number of definitions can be found to describe Software Defined Radio, also known as Software Radio or SDR. The SDR Forum, working in collaboration with the Institute of Electrical and Electronic Engineers (IEEE) P1900.1 group, has worked to establish a definition of SDR that provides consistency and a clear overview of the technology and its associated benefits. Simply Software Defined Radio is defined as:

"Radio in which some or all of the physical layer functions are software defined" [10].

A software defined radio is defined as a form of transceiver in which ideally all aspects of its operation are determined using versatile, general-purpose hardware whose configuration is under software control. The idea of implementing radio functions in software rather than hardware had already been established when, in 1991, Joseph Mitola coined the term ‘Software Radio’ [5]. He said "*A software radio is a radio whose channel modulation waveforms are defined in software. That is, digital waveforms are generated and converted to analog form using a Digital to Analog Converter (DAC) at the transmitter and at the receiver, they are converted to digital form using Analog to Digital Converter (ADC) and demodulated using the software*" [5]. SDR Technology was originally conceived as a means to facilitate better communications between the different forces of the US military. SDR was seen as a technology that could facilitate the interworking of all different radios and radio systems used within the forces. The aim was to eventually reduce the number of different radio systems used by the different military services.

2.3 SDR Advantages

The ability of SDRs to change its physical behavior provides it with several advantages over its hardware-defined counterpart. Primarily, SDRs can be easily modify and implement different physical layer radio protocols unlike hardware radios. By merely editing some code, the designer can change the functionality of a radio system without having to physically change a hardware configuration [11]. This adaptability is useful for several reasons.

For one, a SDR system can be quickly changed to support different hardware protocols. This could eventually be used in a system like cellular phones that need to support several different radio protocols. Instead of

needing a separate module for each protocol, it would merely need one hardware module with different software installed for each necessary radio protocol. Additionally, developers would be able to quickly edit and update their radio system by changing code rather than having to develop and replace hardware modules. This modification functionality could decrease the physical complexity, size, and cost of radio networks by having one device perform multiple functions.

A second advantage of SDRs is that they could be cheaper than dedicated hardware radios in some respects. With hardware radios, any time a radio system needs to be updated or edited, a completely new circuit board must be created which can cost a lot of money if a company has many radios on the market. On the other hand, SDRs would merely need a software update to have additional or improved functionality. Companies would benefit from having the ability to quickly change designs by changing some lines of code rather than changing physical components. This reduces cost by eliminating the need for new physical components when upgrading radio units. The lower cost of SDR devices in comparison to hardware defined radio devices when changing radio systems could drive more consumers and developers to use SDRs in the future.

The ease of testing and implementation of communication standards presents a third advantage of SDRs over hardware-defined radios. First of all, when a new wireless communications protocol is being developed, many tests are needed to determine the standards and specifications of the protocol. With hardware radio systems, new circuits must be designed and created for every test. Then, when changes need to be made, new hardware needs to be purchased. Conversely, with SDR systems, testing and implementation would be simpler, cheaper, and quicker. When testing, code could be changed to test a new specification. This would allow

researchers and developers a very good test-bed for wireless communication systems.

In addition to overcoming some of the limitations of hardware radios, SDRs have potential for functionality not implementable with hardware radios. For example, a cognitive radio is able to analyze the wireless spectrum in an area and adjust its parameters to allow more efficient use of the wireless spectrum to take place in the area. Hardware radios, unable to change their physical protocol, have no hope in ever being able to implement cognitive radios. Consequently, the idea of a fully realized cognitive radio has developed in conjunction with research into software radios. In fact, creating fully functional and robust cognitive radios is one of the main goals of SDR research.

2.4 Receiver architecture

There are many radio receiver architectures to convert RF frequency to IF or baseband, but the most popular among them are the Heterodyne and the Homodyne architecture.

2.4.1 Heterodyne receiver

Heterodyne architecture was invented by Armstrong in 1917, which is the most widely used architecture in wireless transceivers so far. It is dual conversion architecture as illustrated in Figure 2-1.

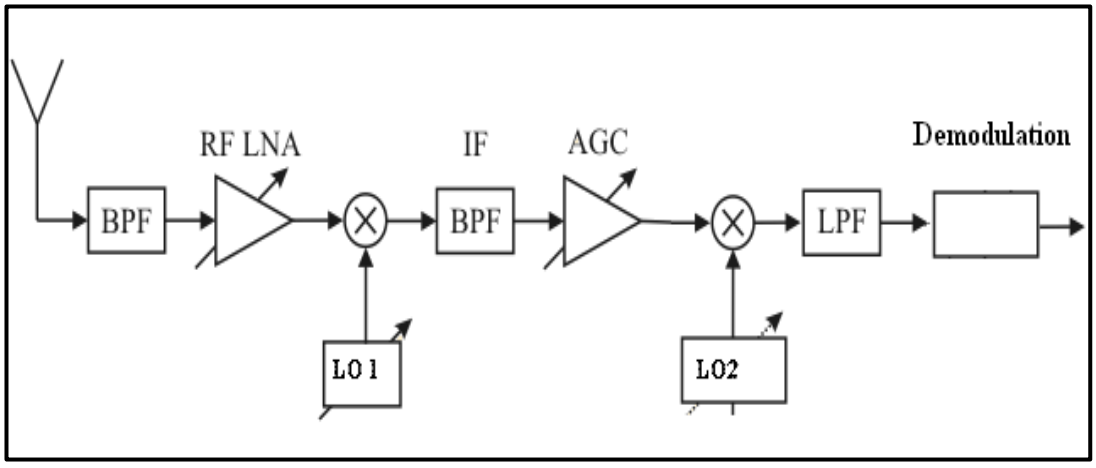


Figure 2-1: Heterodyne receiver architecture.

In heterodyne architectures, the signal band is translated to much lower frequencies so as to relax the Q point required of the channel select filter illustrated in Figure 2-2. The translation is carried out by means of a mixer. To bring the center frequency from w_1 to w_2, the signal is first mixed with a sinusoid $A_0 \cos w_0 t$, where $w_0 = w_1 - w_2$, thereby yielding a band around w_2 and another around $2(w_1 - w_2)$, a lowpass filter then removes the latter. This operation is called "down conversion mixing." Because of its typically high noise, the down conversion mixer is preceded by a low noise amplifier Figure 2-3. The sinusoid is generated by the local oscillator and its frequency is $w_{Local\ OSC} = w_0$.

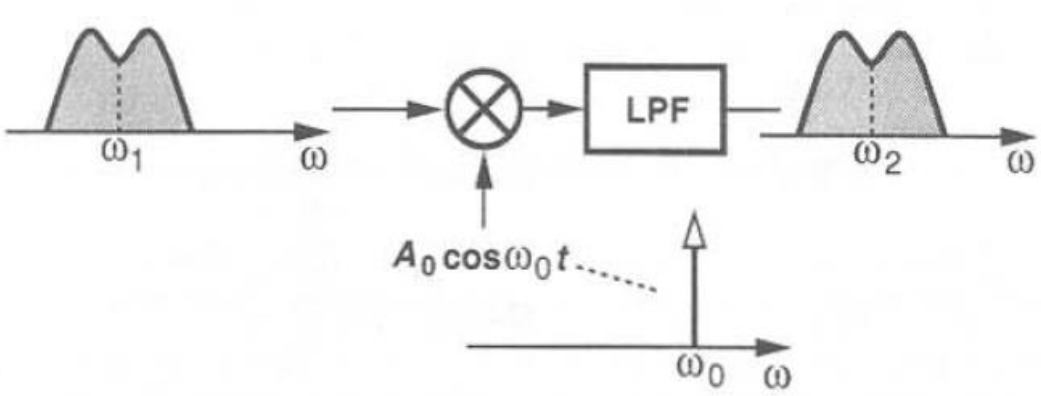


Figure 2-2: Simple heterodyne down conversion.

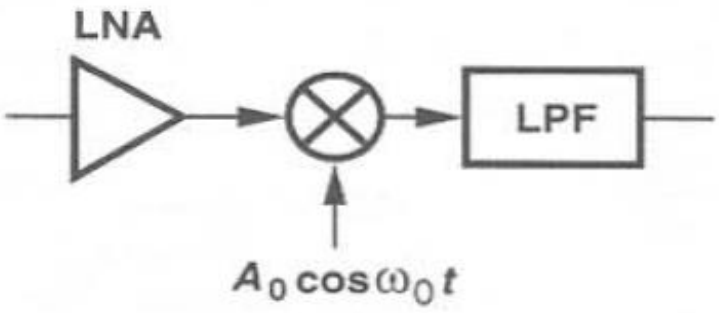


Figure 2-3: Inclusion of an LNA to lower the noise figure.

When choosing the frequencies of the local oscillator w_{LO} and the intermediate frequency IF, in principle one has to take into consideration the image frequency. Note that a simple analog multiplier does not

preserve polarity of the difference between its two input frequencies, i.e. for $x_1(t) = A_1 \cos w_1 t$ and $x_2(t) = A_2 \cos w_2 t$ the lowpass filtered product of $x_1(t)$ and $x_2(t)$ is of the form $\cos(w_1 - w_2)t$, and not different from $\cos(w_2 - w_1)t$. Thus, in the heterodyne architecture, the bands symmetrically located above and below w_{LO} frequency are down converted to the same center frequency, Figure 2-4. If the received band of interest is centered around $w_1 = w_{LO} - w_{IF}$, then the image around $2w_{LO} - w_1 \rightarrow w_{IM} = w_{LO} + w_{IF}$ and vice versa.

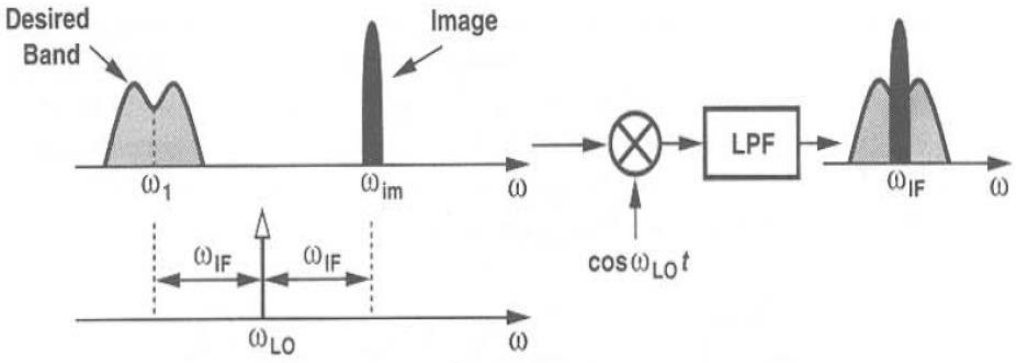


Figure 2-4: Problem of image in heterodyne reception.

The most common approach to suppressing the images is through the use of an image-reject filter placed before the mixer. As depicted in Figure 2-5, the filter is designed to have a relatively a small loss in the desired band and a large attenuation in the image band two requirements that can be simultaneously met if $2w_{IF}$ is sufficiently large.

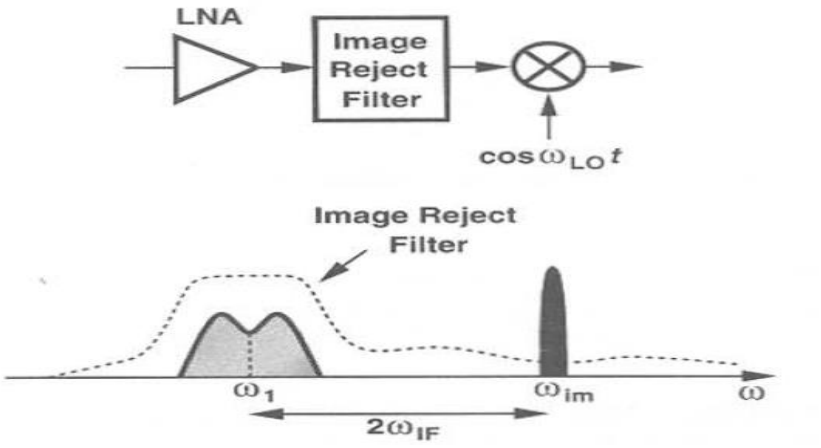


Figure 2-5: Image rejection filter.

2.4.2 Homodyne receiver

In homodyne receiver the RF spectrum is directly translated to the baseband called “direct conversion” or “zero IF”. Figure 2-6 is a simple homodyne receiver, where w_{LO} is equal to the input carrier frequency. Note that the channel selection requires a lowpass filter with relatively sharp cut off characteristics.

For frequency and phase modulation signals, the down conversion must provide quadrature outputs, Figure 2-7, so as to avoid loss of information. This is because the two sides of FM or QPSK spectra carry different information and must be separated into quadrature phases in translation to zero frequency.

The simplicity of homodyne architecture offers two important advantages over a heterodyne. First, the problem of images is circumvented because $w_{IF} = 0$. As a result, no image filter is required and the LNA need not drive a 50 ohms load. Second, the IF SAW filter and subsequent down conversion stages are replaced with lowpass filters and baseband amplifiers.

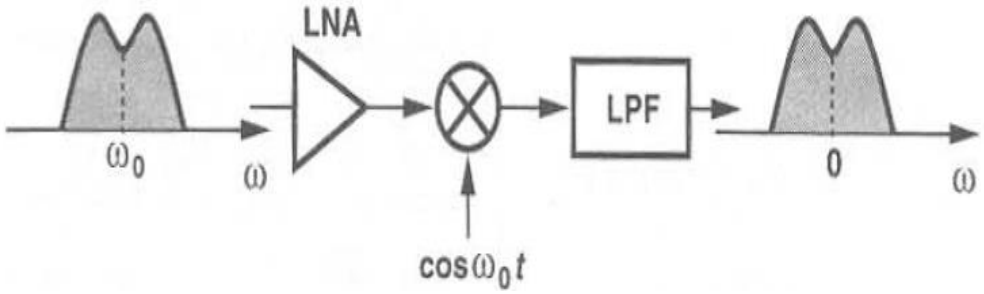


Figure 2-6: Simple homodyne receiver.

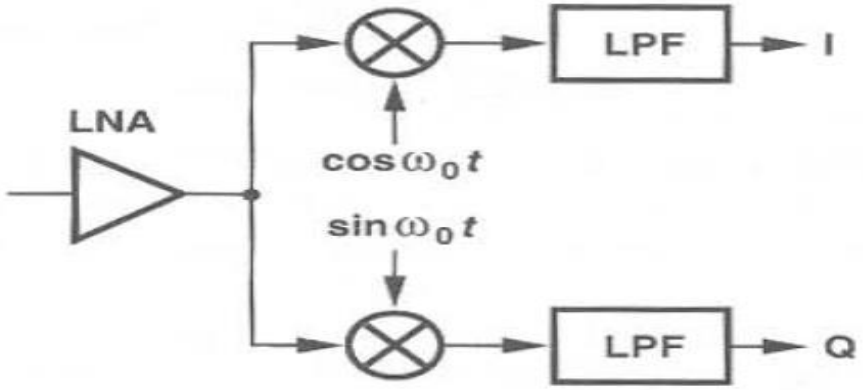


Figure 2-7: Homodyne receiver with quadrature down conversion.

2.4.3 SDR Architecture

A software defined radio consists of, for the most part, the same basic functional blocks as any digital communication system. Software defined radio lays new demands on many of these blocks in order to provide multiple bands, multiple service operation and re-configurability needed for supporting various air interface standards. To achieve the required flexibility, the boundary of digital processing should be moved as close as possible to the antenna, and application specific integrated circuits, which are used for baseband signal processing, should be replaced with programmable implementations.

First generation ‘digital radios’ appeared back in the mid-1990s [12]. As illustrated in Figure 2-8 (a), the analogue part of this radio architecture down convert signals from their RFs to an IF using a Local Oscillator (LO), and then, using a second LO, further down convert the IF signal to

baseband. The baseband signal was then sampled and digitized using an ADC (at a rate of no more than a few 10s of kHz), and then DSP was used to perform the final processing stages to recover the transmitted information.

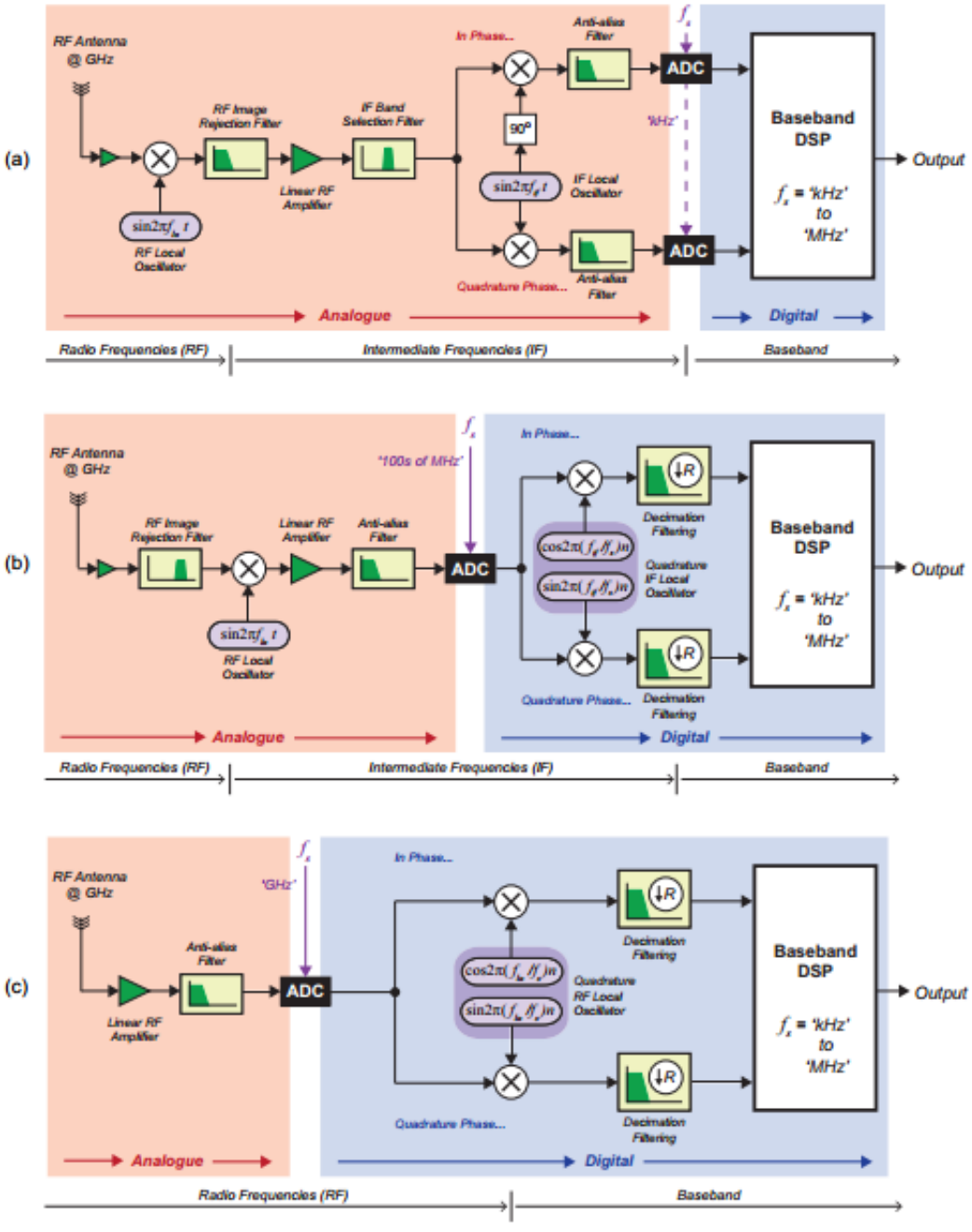


Figure 2-8: The evolution of SDR as the sampling rate of the ADCs increase and they move closer to the antenna:
(a) The baseband digital radio evolves to (b) the IF digital radio, and then to (c) the RF (zero-IF) digital radio [12].

In the next generation of digital radios, which emerged in the 2000s, the sampling and digitization processes started to be performed in some devices at IFs. IFs of around 40 MHz (for example), could be supported by an ADC that sampled at 120 MHz. The first DSP stage of this architecture involved using a Direct Digital Downconverter (DDC) to shift IF signals to baseband using demodulation and decimation filtering, as shown in Figure 2-8 (b). Further DSP processing was then performed once the signal was at baseband. In this architecture, more functionality was implemented in the digital domain, giving greater flexibility for SDR. Ultimately the move has been made to sample RF signals directly as illustrated in Figure 2-8 (c), and down convert them from RF frequencies to baseband in a single stage, using DSP. This is possible today because we are now able to sample in the order of GHz. This is a shift towards the 'Ultimate SDR' architecture presented in Figure 2-9.

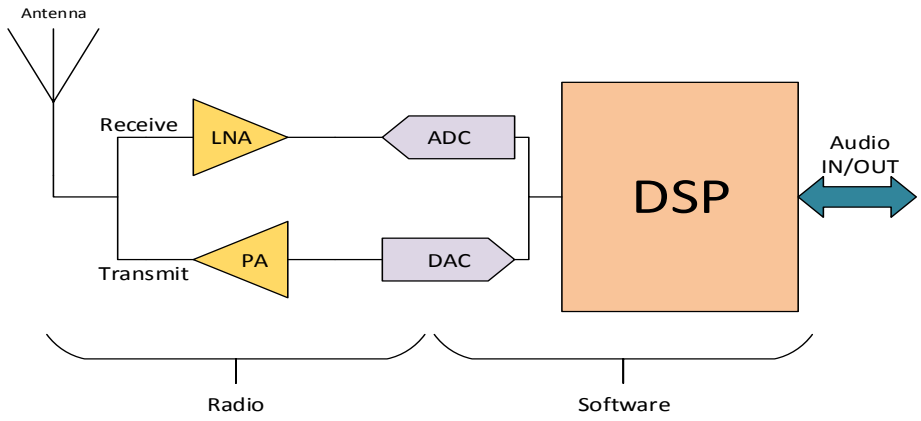


Figure 2-9: An ideal ‘Ultimate’ software defined radio structure.

2.5 SDR Platforms

Real time software defined radio design can be implemented using a variety of digital hardware namely:

a) Field Programmable Gate Arrays (FPGA).
b) Digital Signal Processors (DSP).
c) Application Specific Integrated Circuits (ASIC).
d) General Purpose Processors (GPP).

The different implementation platforms are shown in Figure 2-9. All the four platforms shown in Figure 2-9 possess a level of reprogrammability or reconfigurability (i.e., the ability to modify the hardware or software). The DSP platform is essentially a microprocessor based system optimized for digital signal processing applications. DSPs can be programmed repeatedly with a high level language such as C, MATLAB. Modifications and upgrades to the design are made through these high level languages, thus reducing the design times for each iteration. The flexibility offered by the digital signal processor comes at the cost of efficiency. When there are several computations to be performed, parallel executions of these computations will slow down the rate at which data is processed and this leads to the use of more than one DSP. This solution is limited since synchronizing several DSPs is difficult.

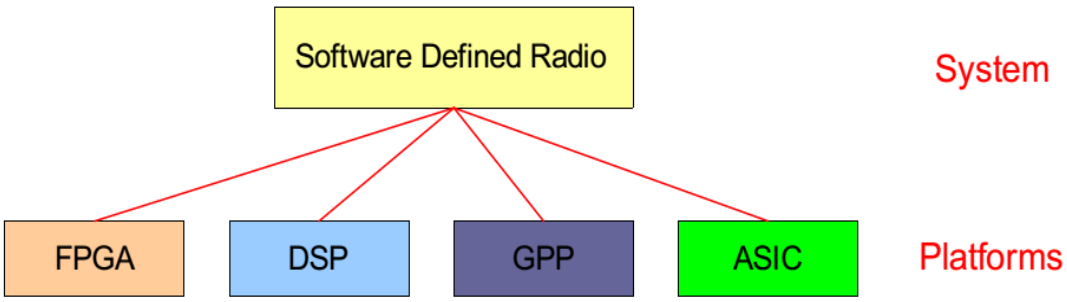


Figure 2-10: Hardware implementations for SDR

A field programmable gate array is a general purpose integrated circuit invented commercially by Xilinx co-founders *Ross Freeman* and *Bernard Vonderschmitt* in 1985 [13], which can be programmed by the designer rather than the device manufacturer. A unique feature of FPGA is that it can be reprogrammed, even after it has been deployed into a system. Field programmable gate array is programmed by downloading a configuration program (bitstream) into the static on-chip random access memory [14]. This is similar to the object code of a microprocessor, in which the bitstream is the product of compilation tools that translate the high level

abstractions produced by a designer into equivalent but low level executable code.

Field programmable gate arrays were designed for multilevel circuits to handle complex circuits on a single chip. Since they are reprogrammable, their configurations can be easily changed to upgrade systems or correct system bugs, making it ideal for prototyping. Field programmable gate arrays are now used in various configurations, as in multimode systems, and are very useful in meeting the needs of a software defined radio implementation.

Application specific integrated circuits ASICs implement the system circuitry in fixed hardware, resulting in the most optimized implementation of the circuit in terms of speed and power consumption. However, ASIC design requires sophisticated circuit design and layout software tools. Also, as the name implies, their use is for specific application and not subject to modification at a later date.

A general purpose processor is similar to DSP as a hardware platform in the design of software defined radio. Like DSP, it offers flexibility and ease of design. Radio functionalities can be implemented in high level languages such as C and C++. Designers can use the familiar approaches of object oriented programming and debugging to develop real time software radio systems. This increases productivity significantly and thus reduces system development time, Table 2-1 summarize differences between hardware architectures [15].

Table 2-1: Comparison of hardware architectures

Device	Parallelism	Gate Reuse and Time Sharing	Programmability	Power usage	speed
GPP	Some	High	High	High	Moderate
DSP	Some	Moderate	High	Moderate	Moderate
FPGA	High	Some	Moderate	Moderate/High	High
ASIC	High	None	Low	Low	High

Digital signal processor is the most generalized type of hardware that can be programmed to perform various functions, while ASIC is the most specialized and can be used only in specific application. Field programmable gate arrays offers a compromise in flexibility between ASIC and DSP platforms.

In general, these hardware components constitute design spaces that trade flexibility, processing speed, and power consumption among other things. There should be a tradeoff between the maximum flexibility and high power consumption of DSP platforms to minimum flexibility and less power consumption of ASICs compared to FPGAs, which have good hardware optimization Table 2-2.

Table 2-2: Digital hardware trade-offs matrix

Design Parameter	weights	GPP	High performance DSP	Low-power DSP	FPGA	ASIC
Performance	%	5	7	3	9	10
Flexibility	%	8	8	8	6	1
Power	%	1	3	7	2	10
Parallelism	%	3	4	1	10	10
Development time	%	6	7	7	4	2
Upgradability	%	7	6	6	5	0

Recently, FPGAs have become increasingly popular due to their ability to reduce design and development cycle time. Furthermore, latest FPGAs

come with intellectual property (IP) cores, which are used for specific applications.

There are other advantages of using FPGAs instead of DSPs for signal processing in commercial telecommunication systems. The power consumption is lower, the size is smaller, quicker to use and the costs are much lower in comparison to DSPs. Since the chip can be reused after fixing the bugs or upgrading a system, they are ideal for prototyping and testing the circuit design. Since FPGAs are reprogrammable, one chip can be configured to perform more than one function and the configurations can be changed during run time.

2.6 FM Theory

Frequency modulation (FM) is a type of angle modulated signal. In frequency modulated systems, the amplitude of the carrier is kept constant while the frequency is changed as a function of the amplitude of the information signal. Thus, the amplitude of the modulating signal is carried by the instantaneous frequency of the modulated carrier, while the frequency is carried in the rate of change of the instantaneous frequency.

A conventional angle modulated signal is defined by the following equation:

$$S_{FM/PM}(t) = A_c \cdot \cos(w_c t + \theta(t)) \quad (2.1)$$

where the instantaneous phase $\emptyset_i(t)$ is defined as:

$$\emptyset_i(t) = w_c t + \theta(t) \qquad (2.2)$$

and the instantaneous frequency $w_i(t)$ of the modulated signal is defined as:

$$w_i(t) = \frac{d}{dt}[w_c t + \theta(t)] = w_c + \frac{d}{dt}[\theta(t)] \quad (2.3)$$

The functions $\theta(t)$ and $\frac{d}{dt}[\theta(t)]$ are referred to as the instantaneous phase and frequency deviations, respectively.

The phase deviation of the carrier $\emptyset(t)$ is related to the baseband message signal $s(t)$. Depending on the nature of the relationship between $\emptyset(t)$ and $s(t)$ different forms of angle modulation can be made.

$$\frac{d}{dt}[\theta(t)] = k_f\, s(t) \qquad (2.4)$$

$$\theta(t) = k_f \int_{t_0}^{t} s(\tau)\, d\tau \qquad (2.5)$$

where k_f is a frequency deviation constant, (expressed in (radian/sec) / volt). It is usually assumed that $t_0 = -\infty$ and $\emptyset(-\infty) = 0$.

Combining equations (2.4) and (2.5) with equation (2.1), we can express the frequency modulated signal as:

$$S_{FM}(t) = A_c\, .cos(w_c t + k_f \int_{-\infty}^{t} s(\tau)\, d\tau\,) \quad (2.6)$$

Chapter Three

FM RECEIVER ARCHITECTURE

Chapter Three

3 FM Receiver architecture

3.1 Overview

In radio communication system, the signal with high frequency has been transmitted. On one hand, only the signal with high frequency can be transmitted over a long distance; on the other hand, the height of antenna has a strong relationship with the signal frequency. The lower the frequency is the higher the antenna is. Thus, to transmit low frequency signal may require a very high antenna which even cannot be made out. Whenever the signal with low frequency needs to be transmitted, it is necessary and important to modulate it to a high frequency signal. Only in this way the signal can be transmitted in the radio communication system. The signal used to carry message is called carrier signal; typically, it is a high-frequency sinusoid or cosine waveform. The carrier signal can be transmitted via the air over a long distance. The process of making the radio frequency carrier signal carry the information signal with low frequency is modulation. Modulation can be realized by varying one or more features of a carrier signal.

When the receiver receives modulated signal, it has to process the modulated carrier signal and get the original information; this process is demodulation. Its function is opposite to that of modulation.

3.2 FM Modulator and Demodulator for Single Tone

Modulation is the process of varying one or more features of a carrier signal. If the modulating signal is analog and the variation for the parameters of carrier signal based on the modulating signal is continuous, the modulation is treated as analog modulation.

The parameters can be changed in carrier signal are amplitude and angle, while the angle contains frequency ω and phase θ. When the amplitude of carrier signal varies as the modulating signal, it is amplitude modulation (AM); if the other two parameters are changed, it is called frequency modulation (FM) and phase modulation (PM) respectively.

3.2.1 FM transmitter

Since frequency modulation is a nonlinear process, an exact description of the spectrum of a frequency-modulated signal for an arbitrary message signal is more complicated than linear process. However, if $s(t)$ is sinusoidal, then the instantaneous frequency deviation of the angle-modulated signal is sinusoidal and the spectrum can be relatively easy to obtained. Figure 3-1 shows a single tone ($s(t)$ message signal), a carrier frequency and modulated FM signal, represented in time domain.

If we assume $s(t)$ to be sinusoidal then:

$$s(t) = A_m \cos(w_m t) \qquad (3.1)$$

The modulated signal can be expressed as

$$S_{FM}(t) = A_c \cdot \cos(w_c t + A_m k_f \int_{-\infty}^{t} \cos(w_m \tau)\, d\tau) \qquad (3.2)$$

The integral does not convert so

$$S_{FM}(t) = A_c \cdot \cos(w_c t + A_m k_f \lim_{\tau \to \infty} \int_{t_0}^{t} \cos(w_m \tau)\, d\tau) \qquad (3.3)$$

$$S_{FM}(t) = A_c \cdot \cos(w_c t + \frac{A_m k_f}{w_m} [\sin(w_m t) + \lim_{\tau \to \infty} \sin(w_m \tau)]) \qquad (3.4)$$

$$S_{FM}(t) = A_c \cdot \cos(w_c t + \frac{A_m k_f}{w_m} [\sin(w_m t)]) \qquad (3.5)$$

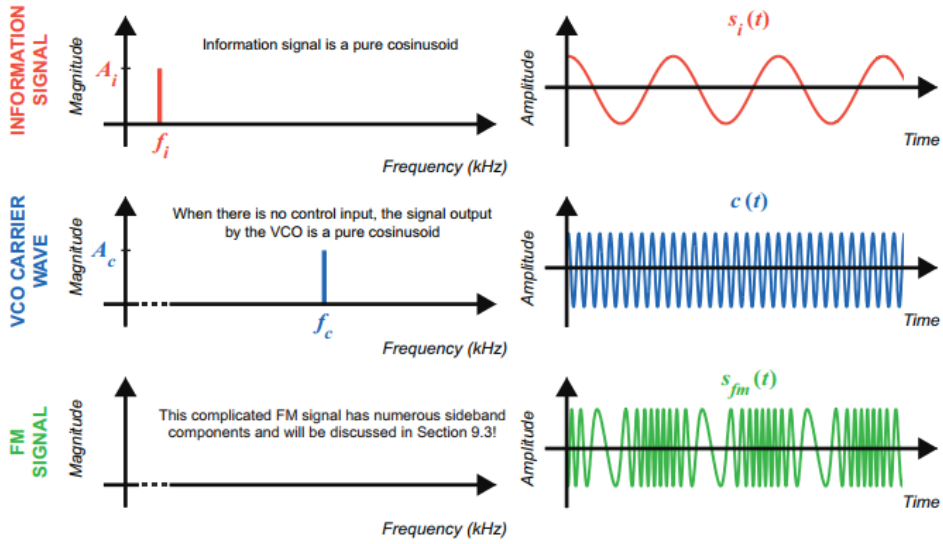


Figure 3-1: Time domain plots demonstrating a single tone information signal being FM modulated

For FM modulation:

$$\beta = \frac{A_m\, k_f}{w_m} \qquad (3.6)$$

where β is called the modulation index which defined only for sinewave modulation and it represents the maximum phase deviation produced by the modulating signal.

$$S_{FM}(t) = A_c\,.cos(\,w_c t + \beta\,[sin(w_m t)]\,) \qquad (3.7)$$

Applying the basic trigonometry:

$$cos(a+b) = cos\,a\,cos\,b - sin\,a\,sin\,b \qquad (3.8)$$

Then, the FM modulated signal equation will be expressed as:

$$S_{FM}(t) = A_c\,[cos(w_c t)\,\cos(\beta\,[sin(w_m t)]) - sin(w_c t)\,\sin(\beta\,[sin(w_m t)])] \quad (3.9)$$

3.2.2 FM receiver

At the receiver part, the received signal will be multiplied by signal from local oscillator:

$$A_c \,.\, cos(w_c t) \;\; and \;\; A_c \,.\, sin(w_c t) \qquad (3.10)$$

The result obtained:

$$S_{FM}(t) * A_c \,.\, cos(w_c t) =$$

$$A_c{}^2 . cos^2(w_c t)\; cos(\beta\,[sin(w_m t)]) - A_c{}^2 . sin(w_c t)\; sin(\beta\,[sin(w_m t)])\; cos(w_c t) \qquad (3.11)$$

$$S_{FM}(t) * A_c \,.\, sin(w_c t) =$$

$$A_c{}^2 . cos(w_c t)\; cos(\beta\,[sin(w_m t)])\; sin(w_c t) - A_c{}^2 . sin^2(w_c t)\; sin(\beta\,[sin(w_m t)]) \qquad (3.12)$$

Applying the basic trigonometry:

$$\left.\begin{array}{l} sin\,2\theta = 2\,sin\,\theta\;\; cos\,\theta \\ cos^2(\theta) = \frac{1}{2} + \frac{1}{2} cos(2\theta) \\ sin^2(\theta) = \frac{1}{2} - \frac{1}{2} cos(2\theta) \end{array}\right\}$$ on equations (3.11) and (3.12) yields to:

$$S_{FM}(t) * A_c \,.\, cos(w_c t) =$$

$$A_c{}^2 . \left(\frac{1}{2} + \frac{1}{2} cos(2w_c t)\right) \cos(\beta\,[sin(w_m t)]) - A_c{}^2 . \left(\frac{1}{2} sin(2w_c t)\right) sin(\beta\,[sin(w_m t)]) \qquad (3.13)$$

$$S_{FM}(t) * A_c \,.\, sin(w_c t) =$$

$$A_c^2 \left(\tfrac{1}{2} sin(2w_c t)\right) cos(\beta\,[sin(w_m t)]) - A_c^2 \left(\tfrac{1}{2} - \tfrac{1}{2} cos(2w_c t)\right) sin(\beta\,[sin(w_m t)]) \qquad (3.14)$$

$$S_{FM}(t) * A_c \,.cos(w_c t) =$$

$$\frac{{A_c}^2.cos(\beta\,[sin(w_m t)])}{2} + \frac{1}{2}cos(2w_c t)cos(\beta\,[sin(w_m t)]) - \frac{{A_c}^2}{2}sin(2w_c t)\,sin(\beta\,[sin(w_m t)])$$

(3.15)

$$S_{FM}(t) * A_c \,.sin(w_c t) =$$

$$\frac{{A_c}^2}{2}sin(2w_c t)\,cos(\beta\,[sin(w_m t)]) - \frac{{A_c}^2.sin(\beta\,[sin(w_m t)])}{2} - \frac{1}{2}cos(2w_c t)\,sin(\beta\,[sin(w_m t)])$$

(3.16)

Then the two signals pass through low pass filters to give I & Q signals:

$$I = S_{FM}(t) * A_c \,.cos(w_c t) = \frac{{A_c}^2}{2}\,.cos(\beta\,[sin(w_m t)]) \qquad (3.17)$$

$$Q = S_{FM}(t) * A_c \,.sin(w_c t) = \frac{{A_c}^2}{2}\,.sin(\beta\,[sin(w_m t)]) \qquad (3.18)$$

After that, I & Q signals pass to the arctangent function ($\tan^{-1}(\theta)$) to recover the basic modulating audio signal $s(t)$.

$$tan^{-1}\left(\frac{Q}{I}\right) = tan^{-1}\left(\frac{\frac{{A_c}^2}{2}.\sin(\beta\,[sin(w_m t)])}{\frac{{A_c}^2}{2}.\cos(\beta\,[sin(w_m t)])}\right) \qquad (3.19)$$

$$tan^{-1}(\tan(\beta\,[sin(w_m t)])) = \beta\,[sin(w_m t)] \qquad (3.20)$$

$$\frac{A_m\,k_f}{w_m}[sin(w_m t)] = \frac{\beta}{A_m}\,s(t) \qquad (3.21)$$

Demodulation can be summarized in the Figure 3-2 below:

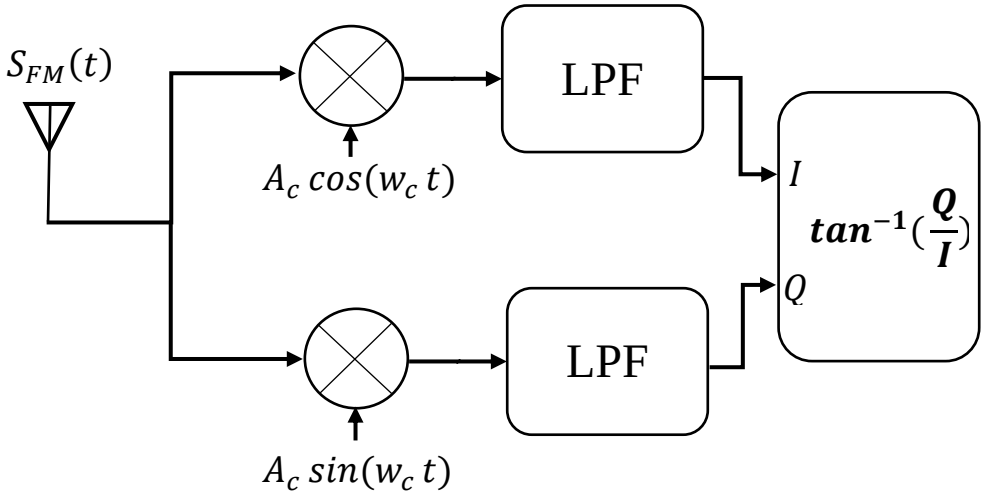


Figure 3-2: FM receiver

3.3 Bandpass sampling theory

As per Nyquist sampling theorem, a signal must be sampled at a rate greater than twice its maximum frequency component in order to ensure unambiguous data. If the Nyquist criterion is not met, aliasing will occur. Say, for example, if the maximum frequency of a sine wave is 70 MHz, then the minimum sampling frequency required is 140 MSPS, as per Nyquist criteria. If we use this Nyquist criterion, that is, the sampling frequency is sufficiently high which will not have any overlapped frequency components in the frequency domain; it is called normal sampling or oversampling. $2 * F_{max}$ is called the Nyquist sampling rate where F_{max} is the maximum frequency component in the signal. Also, the F_{max} is called the Nyquist frequency. Nyquist rate is the minimum sampling rate to avoid aliasing [16].

Under sampling is also known as band pass sampling, harmonic sampling or super-Nyquist sampling. Nyquist-Shannon Sampling theorem, which is the modified version of the Nyquist sampling theorem, says that the

sampling frequency needs to be twice the signal bandwidth and not twice the maximum frequency component, in order to be able to reconstruct the original signal perfectly from the sampled version. The bandpass sampling theorem for real bandpass signals is stated as follows [17]: If a bandpass signal has bandwidth BW and highest frequency f_H , the signal can be sampled and reconstructed using a sampling frequency of $f_s = \frac{2\,f_H}{n}$, where n is the largest integer not exceed $\frac{f_H}{BW}$. All higher sampling frequencies are not necessarily usable unless they exceed $2\,f_H$, which is the value of f_s dictated by the lowpass sampling theorem [18].

The ADC using bandpass sampling can be down convert at low IF or at directly baseband without analog mixer. Therefore, this sampling technique can be applied to the SDR system usefully [19]. Bandpass sampling technique is the intentional aliasing of the bandwidth of the signal. Therefore, the sampling frequency is no longer based on the frequency of the RF carrier, but rather on the bandwidth of RF signal.

In real bandpass sampling, not all the sampling frequency below the Nyquist frequency is allowed, because this sampling technique is occurred aliasing by negative frequency part of self-signal. The available sampling frequency region without aliasing are expressible as:

$$\frac{2\,f_H}{n} \leq f_s \leq \frac{2\,f_L}{n-1} \qquad (3.22)$$

Where $f_s\,, f_L\; and\; f_H$ denote sampling frequency, lower bound and upper bound of signal, respectively. And n is an integer given by:

$$1 \leq n \leq \left|\frac{f_H}{BW}\right| \qquad (3.23)$$

In this project, the receiver operates on the frequency of FM band from $f_L = 87.5$ MHz to $f_H = 108$ MHz. The bandwidth is given by

$$BW = f_H - f_L = 108 - 87.5 = 20.5\,MHz \qquad (3.24)$$

The sampling conditions are satisfied for

$$1 \le n \le \left(\left| \frac{108}{20.5} \right| = 5.27 \right) \qquad (3.25)$$

Therefore, n can be 1, 2, 3, 4, or 5. The value $n = 5$ gives the lowest sampling frequencies interval $43.2\,MHz \le f_s \le 43.75\,MHz$ and this is a scenario of under sampling.

3.4 Frequency demodulation

The concept of measuring the instantaneous frequency $w_i(t)$ of a complex sinusoidal signal is by computing the derivative of the signal's instantaneous phase $\theta(n)$ as shown in Figure 3-3 [20]. This is the traditional discrete-signal FM demodulation method [21], and it works fine [22]. The demodulator's instantaneous output frequency is

$$w_i(n) = w_s(\Delta\theta_{rad}(n)) \quad (3.26)$$

where w_s is the sample rate.

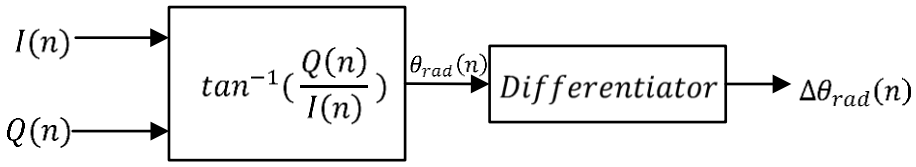


Figure 3-3: Frequency demodulator using an arctangent function.

Computing instantaneous phase $\theta(n)$ requires an arctangent operation, which is difficult to implement accurately without considerable computational resources. Here's a scheme for computing $\Delta\theta(n)$ for use in equation (3.32) without the intermediate $\theta(n)$ phase computation. We derive the $\Delta\theta(n)$ computation algorithm as follows, initially using continuous-time variables based on the following definitions:

$I(t)$: in-phase signal,

$Q(t)$: quadrature phase signal,

$\theta(t)$: instantaneous phase = $tan^{-1}(\frac{Q(t)}{I(t)})$,

$\Delta\theta(t)$: time derivative of $\theta(t)$.

The following algorithm is based on the assumption that the spectrum of the $I(t) + jQ(t)$ signal is centered at zero Hz. First, we let $r(t) = \frac{Q(t)}{I(t)}$ be the signal for which we are trying to compute the derivative of its arctangent. The time derivative of $tan^{-1}r(t)$ a calculus identity, is

$$\Delta\theta(t) = \frac{d}{dt}[\, tan^{-1}r(t)] = \frac{1}{1+r^2(t)}\frac{d}{dt}[r(t)] \quad (3.27)$$

Because $\frac{d}{dt}[r(t)] = \frac{d}{dt}\left[\frac{Q(t)}{I(t)}\right]$, we use the calculus identity for the derivative of a ratio to write

$$\frac{d}{dt}[r(t)] = \frac{d}{dt}\left[\frac{Q(t)}{I(t)}\right] = \frac{I(t)\frac{d}{dt}[Q(t)] - Q(t)\frac{d}{dt}[I(t)]}{I^2(t)} \quad (3.28)$$

Placing Eq. (3.28)'s result into Eq. (3.27):

$$\Delta\theta(t) = \frac{1}{1+r^2(t)}\frac{I(t)\frac{d}{dt}[Q(t)] - Q(t)\frac{d}{dt}[I(t)]}{I^2(t)} \quad (3.29)$$

Replacing $r(t)$ in Eq. (3.29) with $\frac{Q(t)}{I(t)}$ yields

$$\Delta\theta(t) = \frac{1}{1+(\frac{Q(t)}{I(t)})^2(t)}\frac{I(t)\frac{d}{dt}[Q(t)] - Q(t)\frac{d}{dt}[I(t)]}{I^2(t)} \quad (3.30)$$

After multiply the numerator and denominator of the first ratio in Eq. (3.30) by $I^2(t)$ and replace (t) with our discrete time variable index (n) to arrive at our final result of

$$\Delta\theta(n) = \frac{I(n)\frac{d}{dn}[Q(n)] - Q(n)\frac{d}{dn}[I(n)]}{I^2(n)+Q^2(n)} \quad (3.31)$$

The implementation of this algorithm, where the derivatives of $I(n)$ and $Q(n)$ are of $I'(n)$ and $Q'(n)$ respectively, is shown in Figure 3-4 and Figure 3-5. The $\Delta\theta(n)$ output sequence is used in equation (3.26) to compute instantaneous frequency. The Differentiators are tapped-delay line FIR differentiating filters with an odd number of taps.

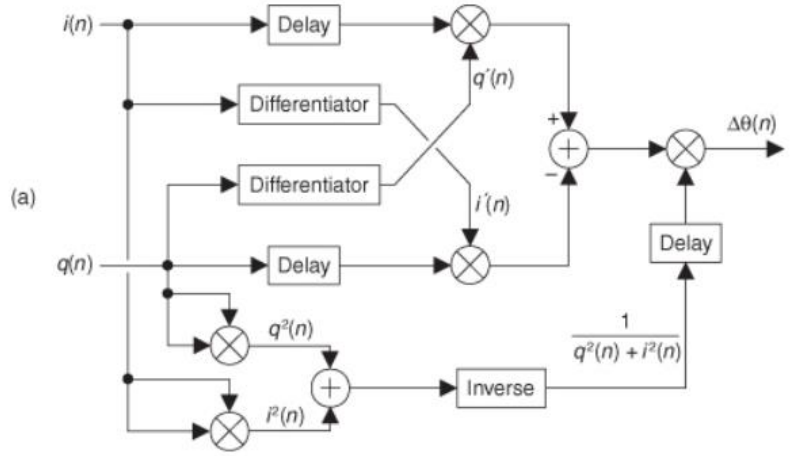


Figure 3-4: Frequency demodulator without arctangent (standard process).

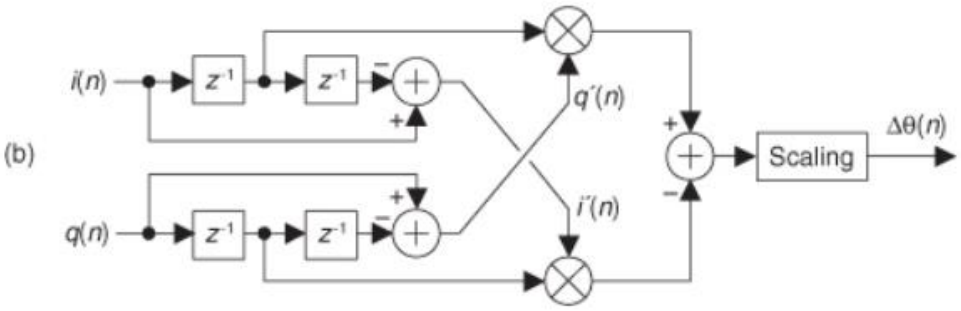


Figure 3-5: Frequency demodulator without arctangent (simplified process).

Chapter Four

IMPLEMENTATION, RESULTS AND DISCUSSIONS

Chapter Four

4 Implementation, Results and Discussions

4.1 Overview

This digital FM receiver was based on simple radio architecture and demodulation method [21], that to focus on the Features given when using reprogrammable hardware to build SDR. A homodyne radio architecture was used to transfer the received FM radio signal to baseband signal. This design supposed to receive normal FM signal span from 87.5 to 108 MHz as well as defined by the International Telecommunication Union (ITU) for region 1 [23]. The demodulator was based on digital arctangent algorithm which was discussed in chapter three. Also, this design can process single tone as well as multi tone audio signal, but for easier comparison of the results, a single tone audio signal will be used and modulated with FM modulator in SIMULINK software then fed to the input port of the FM receiver.

Figure 4-1 shows tools and software used in this project are:

- MATLAB R2016b
- Xilinx Vivado SDK 2017.3
- Zynq 7000 All programable SoC ZedBoard kit [24]
- Computer

The aim of this chapter is to describe the procedures, tools and designs that were employed to implement the project and achieve its objectives. This including detailed description for the FM receiver blocks designed in SIMULINK environment in MATLAB in addition to the steps for generating bit file, which will be loaded in the FPGA to run the hardware co-simulation.

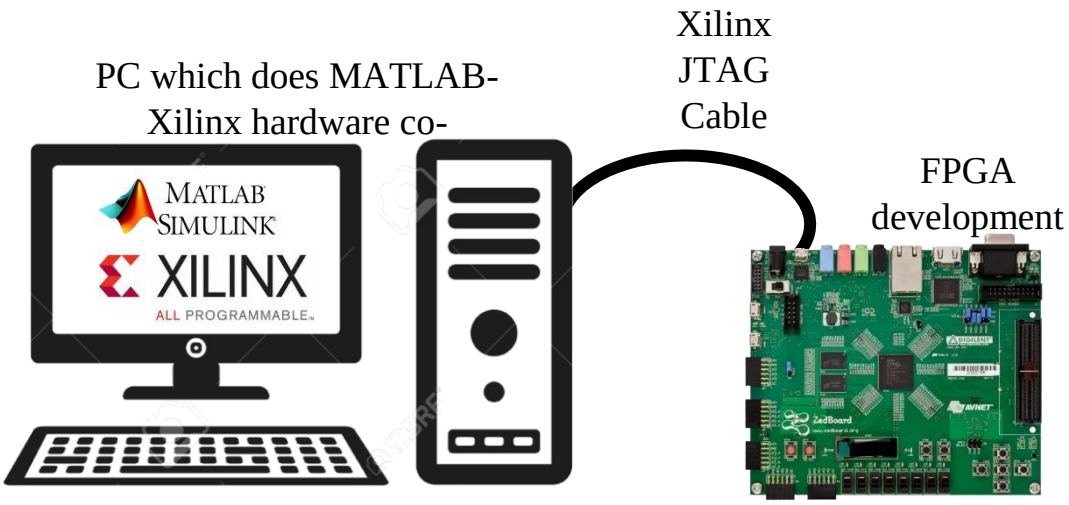


Figure 4-1: PC connected with FPGA development board by JTAG cable to perform hardware cosimulation.

4.2 Design in SIMULINK

SIMULINK was used to design the FM receiver. A single tone audio signal was generated and modulated with FM modulator and fed the input of the receiver. An overall view of the design is shown in Figure 4-2 below.

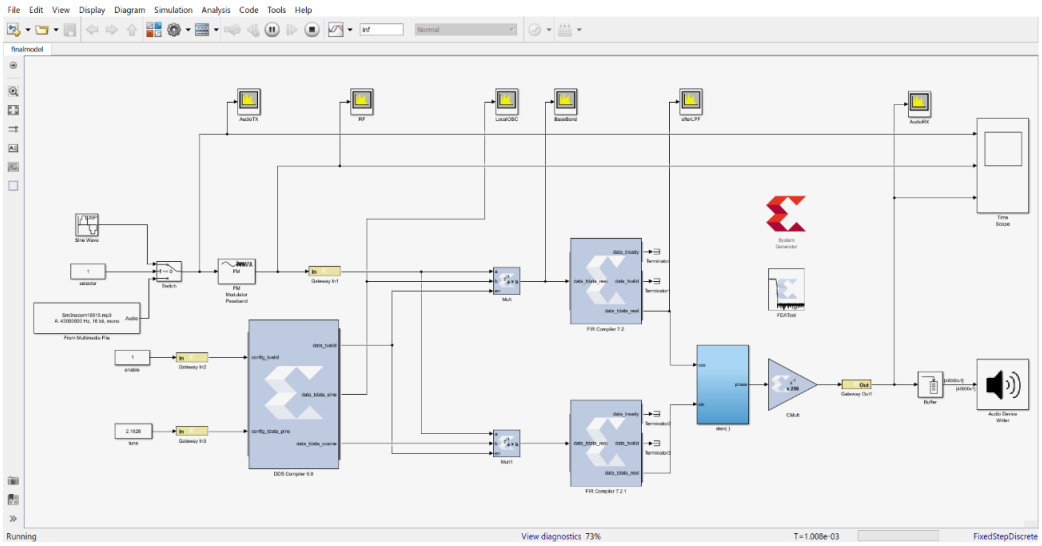


Figure 4-2: Overall view for FM receiver blocks.

The sample rate used was 43 MHz based on under sampling or bandpass sampling technique, which discussed in chapter three.

A GUI was designed in MATLAB to give real time control on parameters for different blocks of this model [25]. Figure 4-3 show GUI window in MATLAB and parameters controlled.

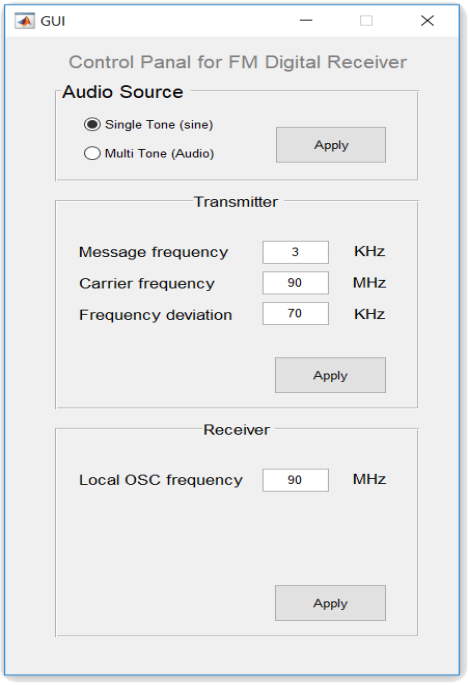


Figure 4-3: Graphical User Interface GUI.

4.2.1 Input signals source

Two types of input signals were used in this model. They can be selected via selector block controlled by GUI from the user. The first signal is a single tone sine wave ranged within voice frequency band from 0.3 to 4 KHz. The second signal is a multi-tone real audio file also can be selected by GUI. Figure 4-4 and Figure 4-5 show block parameters for each type. A time domain graph with SIMULINK time scope and frequency domain graph with SIMULINK spectrum analyzer were shown in Figure 4-7, 4-8 and Figure 4-9.

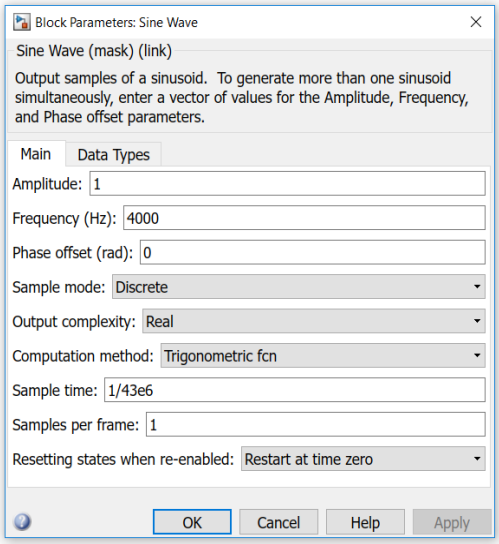


Figure 4-4: Single tone sine wave audio source.

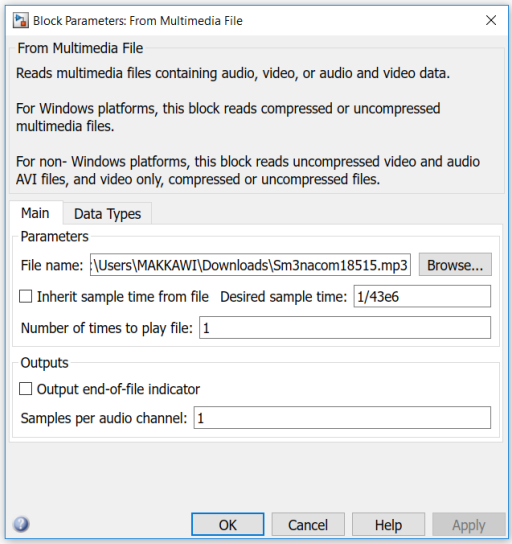


Figure 4-5: Multi tone real voice audio source.

4.2.2 FM modulator

The first process on audio signal is to modulate the signal by FM modulation. This project considering receiver only so a SIMULINK pre-designed FM modulator block was used. FM modulated signal's frequency ranged from 87.5 MHz to 108 MHz with frequency deviation of 70 KHz. Parameters as shown in Figure 4-6 for carrier frequency and frequency deviation, and all can be modified with GUI.

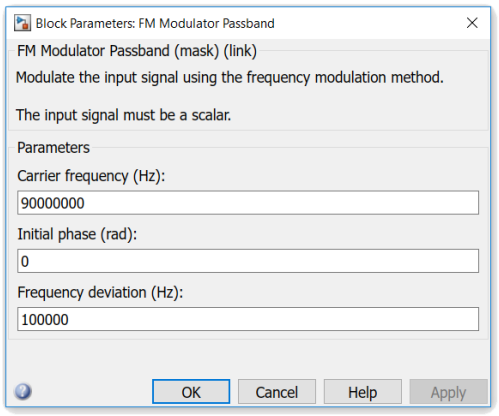


Figure 4-6: FM modulator parameters.

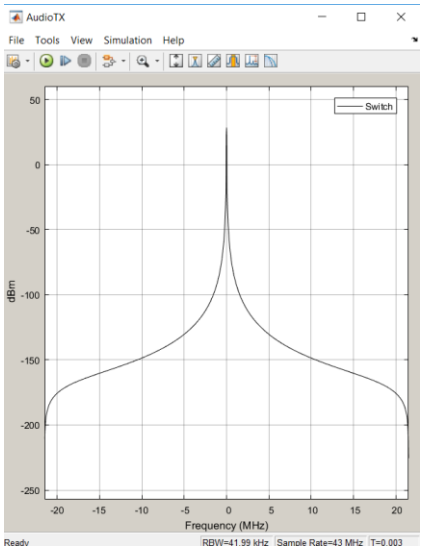


Figure 4-7: Single tone Spectrum.

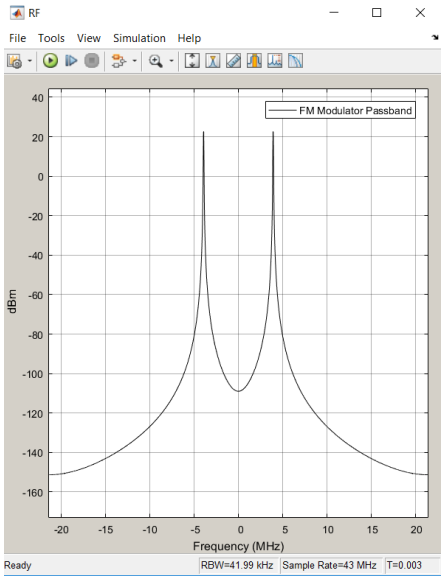


Figure 4-8: Modulated FM signal spectrum

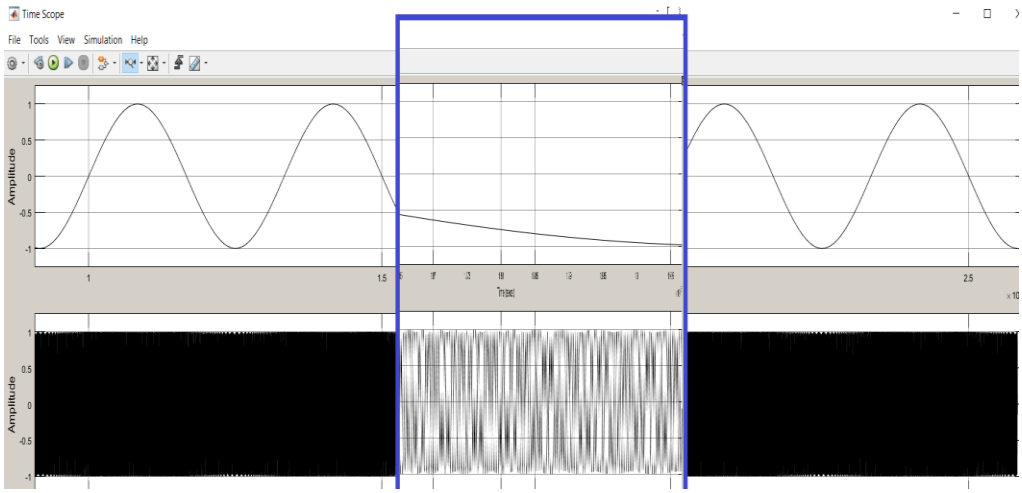


Figure 4-9: Time domain view for single tone sine wave (top) and FM signal modulated with message (bottom)

4.2.3 Direct Digital Synthesizer (DDS)

This block from Xilinx library (DDS compiler 6.0) is responsible of generating digital quadrature signals (sine and cosine) as local oscillator signals that mixed with the incoming channels I & Q. The block parameters are shown Figure 4-10.

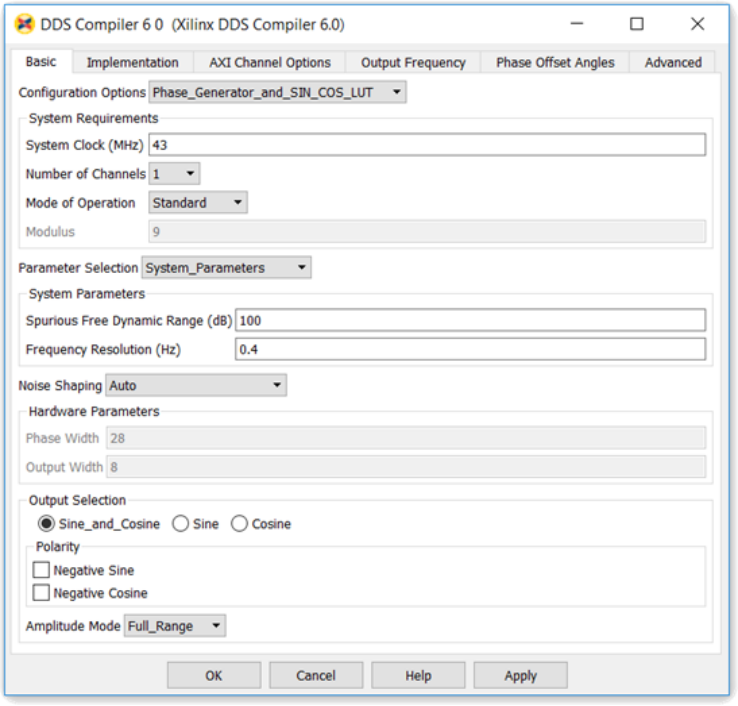


Figure 4-10: DDS block parameters.

Configurations of this parameter [26] allow for two parts of the DDS (the phase generation part and LUT part) to be instantiated separately or together, it was set to phase generator and sin cos LUT together. The main clock feed this block is 43 MHz and the output frequency is controlled with GUI from user. The output signal is shown in Figure 4-11.

4.2.4 Mixer

Mixer Mult & Mult1 are just digital multipliers, they multiply the incoming signal with the output of the DDS cos and sin to get I and Q respectively, and it generates the summation and subtraction of the two signals.

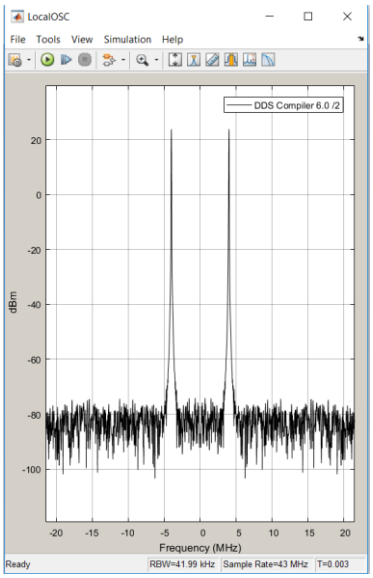


Figure 4-11: DDS output signal.

4.2.5 Filter

In DSP, a lot of techniques used to design digital filters for receivers. Basically, filter is a device to extract the information from the noisy signal. Here, the main purpose of using filter is to remove harmonics and unwanted signals generated from the previous stage (mixers) as discussed in demodulation equations of chapter three. Digital filters are classified into two types, finite duration impulse response (FIR) and infinite duration

impulse response (IIR) filters. In this model, FIR filter was used and implemented with Xilinx FIR compiler 7.2 block [27] Figure 4-12.

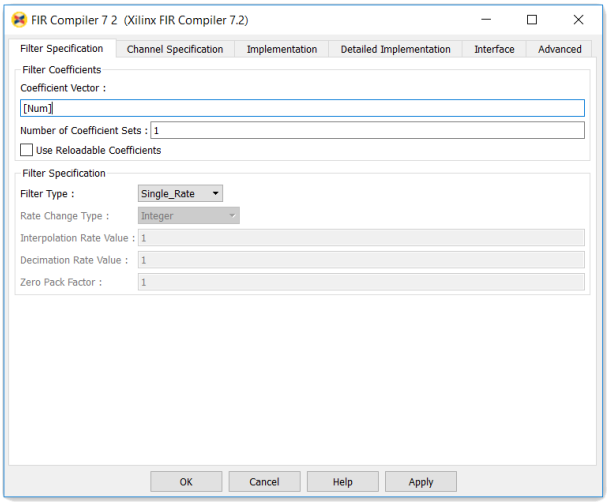


Figure 4-12: Xilinx FIR compiler 7.2

The FDA tool has been used for obtaining the desired filter response (see Figure 4-13) with parameters as mentioned in table 4-1, and then to export designed filter's coefficient to use in Xilinx FIR block [28].

Table 4-1: Low pass filter specifications.

Filter parameters	Value
Structure	Direct FIR filter
Response	Low pass filter
Method	Equiripple
Filter order	Minimum order
Sample frequency	43 MHz
Frequency pass	4 MHz
Frequency stop	5 MHz
Amplitude pass	1 dB
Attenuation	150 dB

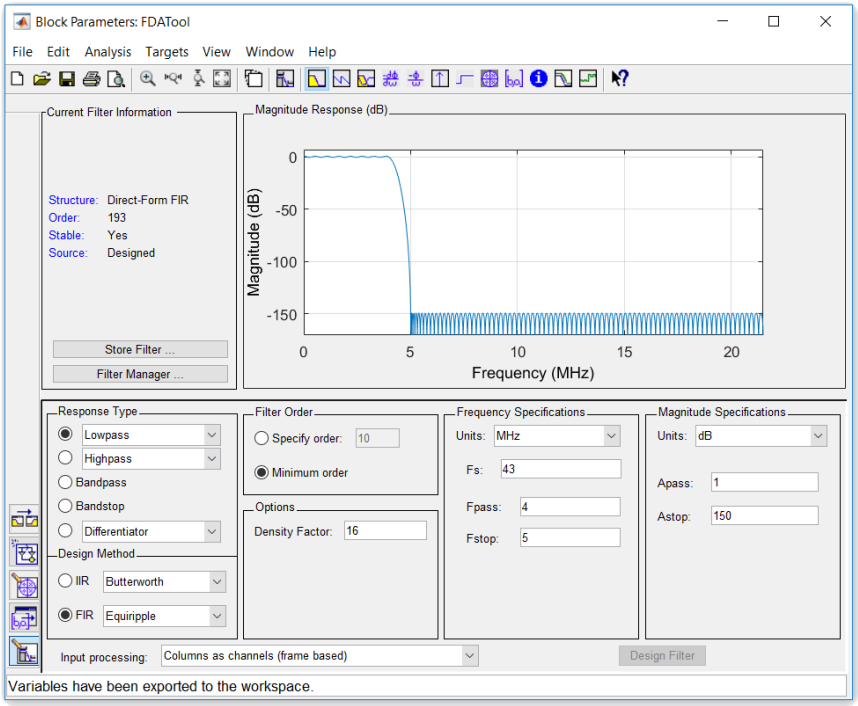


Figure 4-13: FDA tool graphical interface.

4.2.6 Arctangent function

The selected method for FM demodulation was based on extracting the angle of the filtered FM signal from bath I and Q signals (discussed in chapter three). There for, an arctangent function (Figure 4-14) was used, but instead, some equivalent blocks were implemented in SIMULINK to minimize the resources used in FPGA [21].

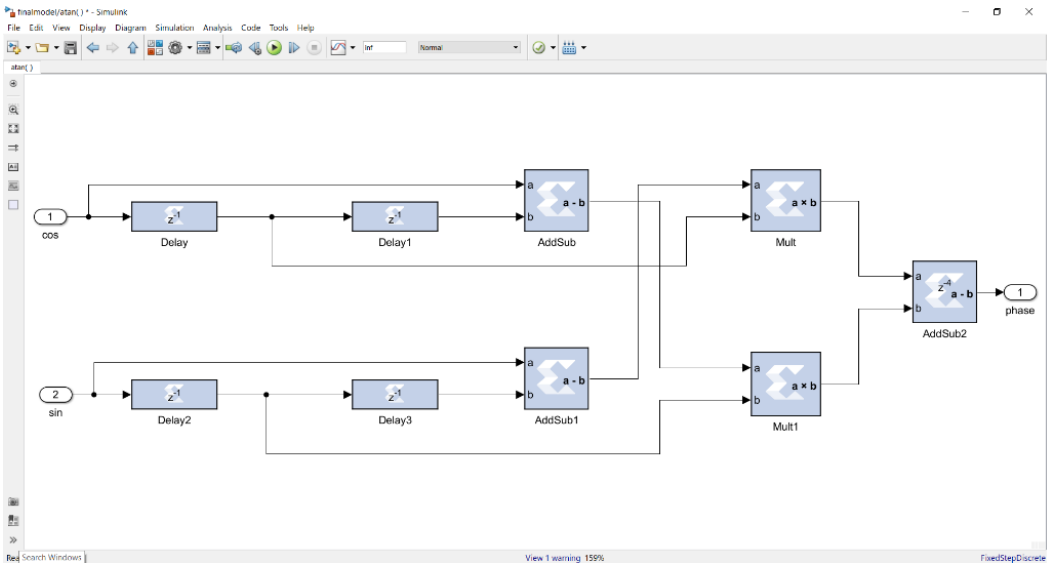


Figure 4-14: Equivalent simplified arctangent function.

4.2.7 Received signal

After extracting the modulated message from FM signal through all previous stages, a final stage is needed to correct the amplitude. A gain block is placed after the arctangent function block. For single tone message it is easy to compare message transmitted with message received on the Time scope Figure 4-15, but for real voice file its difficult Figure 4-16. SIIMULINK can play signals on computer loud speaker by using audio device writer block Figure 4-17. This block was used with buffer to collect group of samples and play it with real time.

From these figures, the demodulated single tone signal or the real audio signal were typical as input one. The delay was according to the processing time in the digital FM receiver. By these results, the digital FM receiver was verified and ready for generation and to be implemented on FPGA kit.

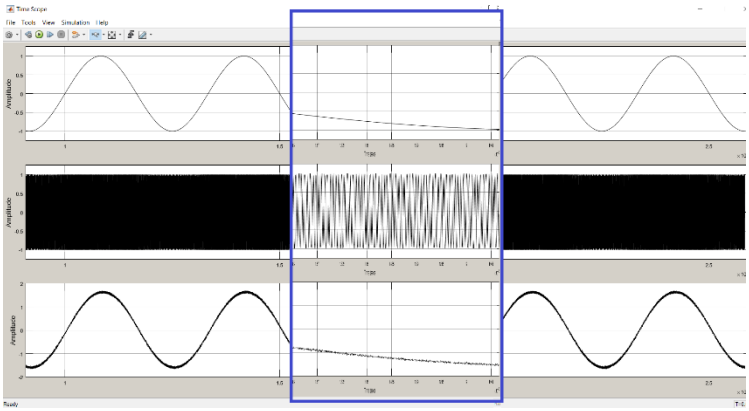


Figure 4-15: transmitted and received single tone message.

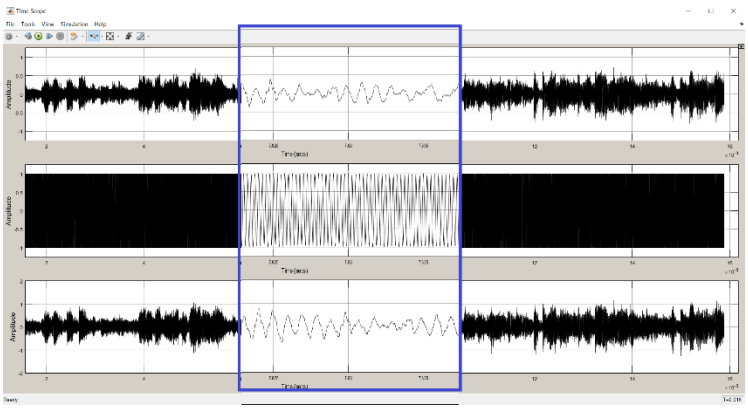


Figure 4-16: transmitted and received real audio (multi tone) message.

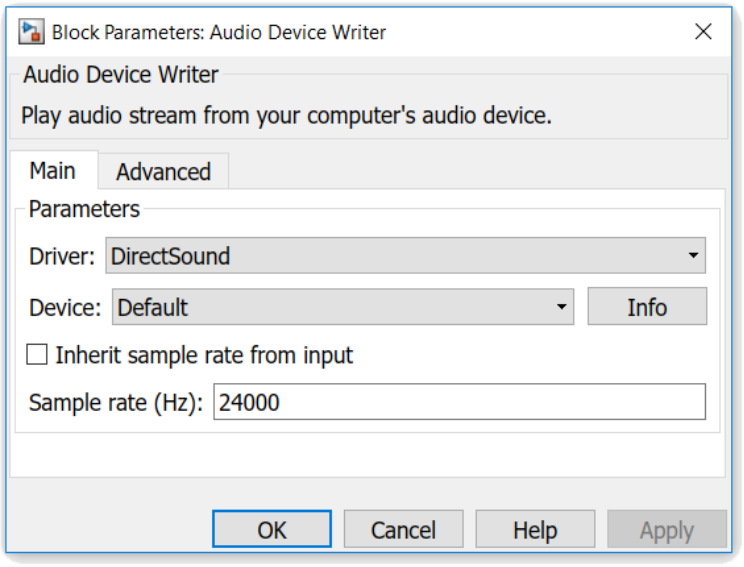


Figure 4-17: Audio device writer parameters.

4.3 Generating bit file and hardware co-simulation

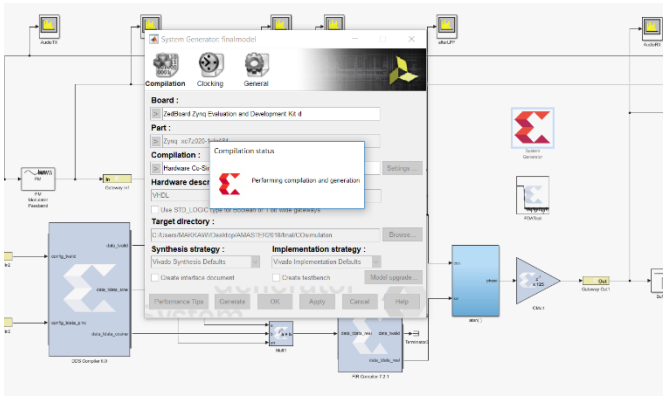


Figure 4-18: During Generating of digital FM receiver JTAG co-simulation block.

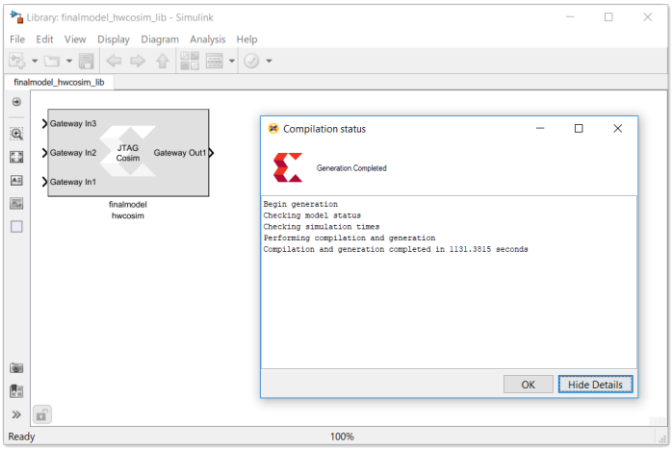


Figure 4-19: After Generating of digital FM receiver JTAG co-simulation block.

The figures above represent the generating of digital FM receiver JTAG co-simulation block (See [29]). This block intended for hardware co-simulation on Xilinx Zynq-7000 FPGA on ZedBoard evaluation kit. The input was from SIMULINK either sine wave or real audio source, the processing on FPGA kit which was configured by JTAG co-sim block and the audio output was shown on the SIMULINK time scope and heard from computer loud speaker device Figure 4-20.

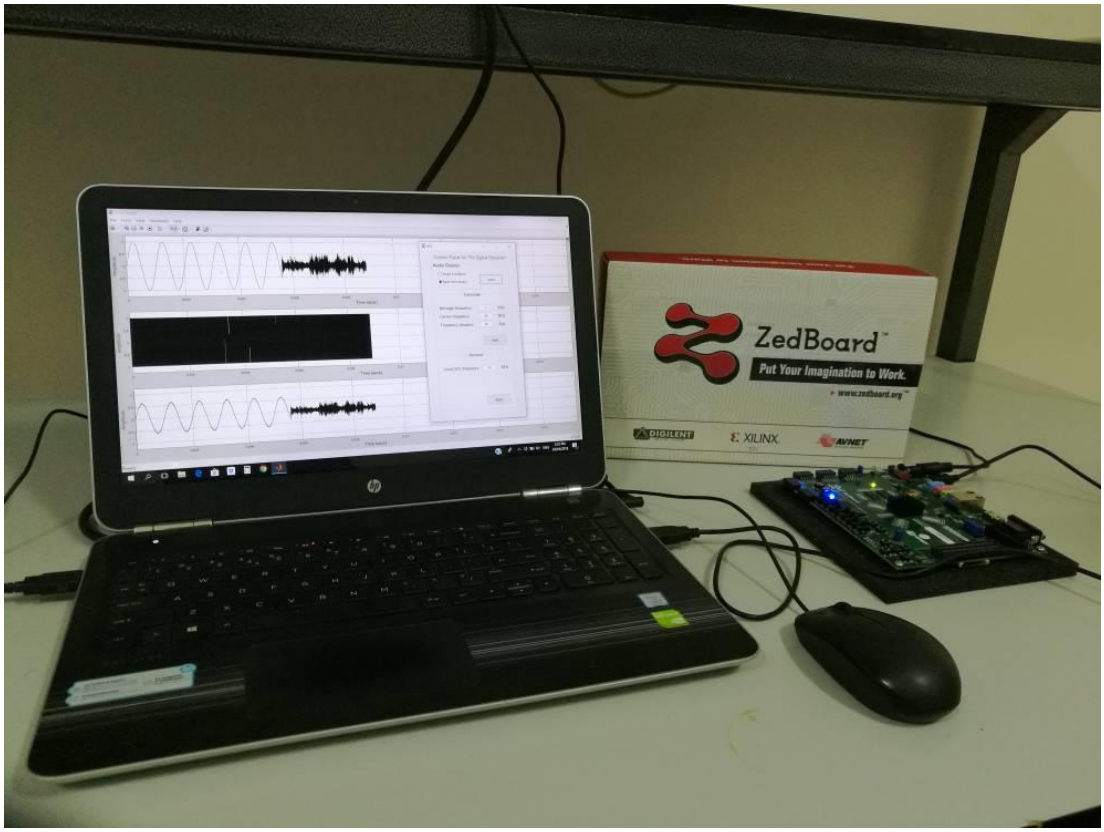


Figure 4-20: Real implementation of hardware co-simulation in laboratory.

When starting digital FM receiver co-simulation, the VIVADO system generator was starting configuring resources of Xilinx Zynq-7000 FPGA on the ZedBoard evaluation kit to implement digital FM receiver on the Kit. A demonstration video for this implementation was recorded in [30].

Chapter Five

CONCLUSIONS AND RECOMMENDATIONS

Chapter Five

5 Conclusions and Recommendations

5.1 Conclusions

Digital implementation of FM receiver can solve some disadvantages found in analog implementation such as complexity, size and high cost. In addition, it can be easier to analyze, reconfigure and test.

In this thesis, digital FM receiver was designed in SIMULINK and implemented successfully on Xilinx Zynq-7000 FPGA on ZedBoard evaluation kit with JITAG co-simulation. The design was tested with both single tone message and multi tone message real voice. Receiver was tunable in real time to select receiving FM channel with GUI designed for this project.

Results achieved were very acceptable and transmitted message was successfully recovered on the receiver which was clear from time domain graphs from time scope and frequency domain graphs from spectrum analyzer.

5.2 Recommendations and Future Works

As an extension to the work carried out in this thesis, the digital FM receiver can be loaded in kit attached with RF unit containing antenna and ADC to receive on the air FM signals and examine the power of Xilinx Zynq-7000 FPGA itself without co-simulation with computer software.

Also, the design can be developed to include other demodulation scheme with friendly use GUI.

Also, more work must be done for FIR filter specially in filter order and processing delay occurred from it.

Vivado design suite software can be the new start for the developed design rather than SIMULINK software because of its very wide control option for each block in design and helpful tool for timing and sample rate matching.

REFERENCES

6 REFERENCES

[1] W. I. Forum, "What is Software Defined Radio," [Online]. Available: http://www.wirelessinnovation.org/assets/documents/SoftwareDefinedRadio.pdf.

[2] G. Youngblood, "A Software-Defined Radio for the Masses, Part 1," QEX, July- August 2002. [Online]. Available: http://www.arrl.org/files/file/Technology/tis/info/pdf/020708qex013.pdf.

[3] D. V. W. L. and f. h. , "How to pack a room of analog FM modulators into a Xilinx FPGA," *Xilinx DSP Magazine,* April 2007.

[4] D. H. N. Abdullah and B. H. A. Hadi, "Design and Implementation of FPGA based Software Defined Radio Using Simulink HDL Coder," *CUAS Journal.*

[5] J. Mitola, Software Radio Architecture–Object Oriented Approaches to Wireless Systems Engineering, Wiley-Interscience, 2000.

[6] S. Gultchev, K. Moessner, D. Thilakawardana, T. Dodgson, R. Tafazolli, S. Vadgama and S. Truelove, Evaluation of SDR Technology, 2006.

[7] S. M. Trimberger, "Three Ages of FPGAs: A Retrospective onthe FirstThirty Years of FPGA Technology," *Proceedings of the IEEE,* vol. 103, no. 3, pp. 318 - 331, 2015.

[8] R. Schiphorst, Demonstration of the Software – Radio Concept, Master thesis, University of Twente, Department Of Electrical Engineering ,Signals & System PO Box 217, 7500 AE Enschede, the Netherlands, June 14, 2000.

[9] A. C. Tribble, "The software defined radio: Fact and fiction," *Radio and Wireless Symposium, 2008 IEEE,* pp. 5-8, 2008.

[10] Software Defined Radio Forum, "SDRF Cognitive Radio Definitions," 8 November 2007. [Online]. Available:

http://www.sdrforum.org/pages/documentLibrary/documents/SDRF-06-R-0011-V1_0_0.pdf.

[11] Michael L. Dickens, Brian P. Dunn, and J. Nicholas Laneman, "Design and Implementation of a Portable Software Radio," *IEEE Communications Magazine*, pp. 58-66, 2008.

[12] R. W. Stewart, K. W. Barlee and D. S. W. Atkinson, Software Defined Radio using MATLAB & Simulink and the RTL-SDR, Scotland,UK: Strathclyde Academic Media, 2015.

[13] Wikipedia, the free encyclopedia, "Field-programmable gate array," [Online]. Available: https://en.wikipedia.org/wiki/Field-programmable_gate_array. [Accessed 5 April 2018].

[14] S. Srikanteswara, "Design and Implementation of a Soft Radio Architecture for Reconfigurable Platforms," in *Virginia Polytechnic Institute and State University*, July 2001.

[15] J. P. Farrell, *Digital Hardware Design Decisions and Trade-offs for Software Radio Systems*, Blacksburg, VA, 2009.

[16] P. Poshala, "Why Oversample when Undersampling can do the Job?," Texas Instruments, 2013.

[17] W. kester, Mixed-signal and DSP Design Techniques, Analog Devices, inc.

[18] P. Cordeiro, R. A. Castro, S. M. Cabral, C. L. Freitas and A. Klautau, "An FPGA-Based Architecture for Digital Filtering at Intermediate Frequency Using Undersampling," Brazil.

[19] M. H. Ali, *Design and Simulation of Digital Down Converter for Software-defined Radio,* M.Sc. thesis, Dept. Electronics Engineering, SUST, Sudan, 2012.

[20] M. A. Padilla, *FM Demodulators in Sofware-Defned Radio Using FPGAs with Rapid Prototyping*, M.Sc. thesis,Brigham Young University - Provo, 2011.

[21] R. G. Lyons, Understanding Digital Signal Processing, Prentice Hall.

[22] K. A. H. ALI, *Design and Implementation of Digital Frequency Modulation Modem Based on Field Programmable Gate Array*, M.Sc. thesis, Dept. Electronics Engineering, UofK, Sudan, 2017.

[23] ITU, "FM / TV Regional Frequency Assignment Plans," [Online]. Available: https://www.itu.int/en/ITU-R/terrestrial/broadcast/Pages/FMTV.aspx. [Accessed 2018].

[24] Zynq™ Evaluation and Development, "Zedboard-Hardware User's Guide," 1 August 2012. [Online]. Available: https://reference.digilentinc.com/_media/zedboard:zedboard_ug.pdf.

[25] J. Park, "MATLAB GUI (Graphical User Interface) Tutorial for Beginners," University of Incheon.

[26] Xilinx, "DDS Compiler v6.0 LogiCORE IP Product Guide," 20 December 2017. [Online]. Available: https://www.xilinx.com/support/documentation/ip_documentation/dds_compiler/v6_0/pg141-dds-compiler.pdf.

[27] "FIR Compiler v7.2 LogiCORE IP Product Guide," 18 November 2015. [Online]. Available: https://www.xilinx.com/support/documentation/ip_documentation/fir_compiler/v7_2/pg149-fir-compiler.pdf.

[28] R. Das, A. Guha and A. Bhattacharya, "FPGA based higher order FIR filter using XILINX system generator," in *Signal Processing, Communication, Power and Embedded System (SCOPES)-2016*, Paralakhemundi, India, 2016.

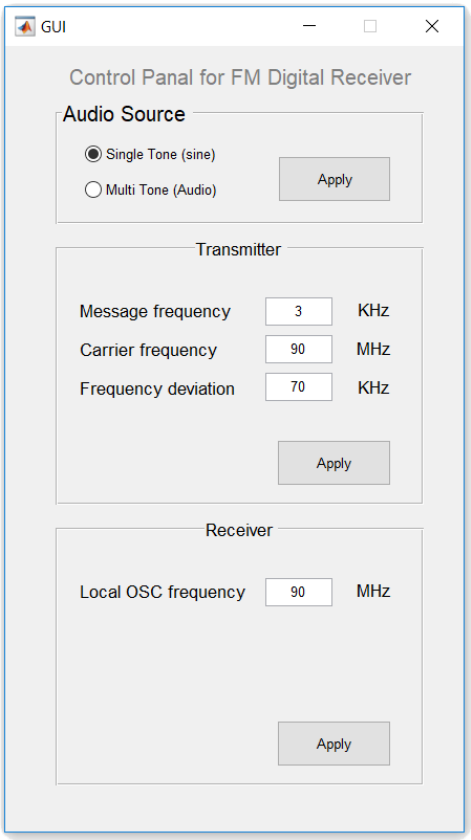
[29] Vivado Design Suite User Guide, "Model-Based DSP Design Using System Generator," 5 April 2017. [Online]. Available: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2018_1/ug897-vivado-sysgen-user.pdf.

[30] M. Makkawi, "Demo video for FM receiver implementation on FPGA," 25 April 2018. [Online]. Available: https://www.dropbox.com/sh/iv2np4apz12u72a/AAA0AyYPFdJpZ3wjAxOisbb8a?dl=0.

APPENDIX

A: MATLAB GUI

GUI window design:



m-file code:

```
function varargout = GUI(varargin)
% GUI MATLAB code for GUI.fig
%      GUI, by itself, creates a new GUI or raises the existing
%      singleton*.
%
%      H = GUI returns the handle to a new GUI or the handle to
%      the existing singleton*.
%
%      GUI('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in GUI.M with the given input arguments.
%
%      GUI('Property','Value',...) creates a new GUI or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before GUI_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to GUI_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help GUI

% Last Modified by GUIDE v2.5 27-Mar-2018 14:31:45

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
```

```
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @GUI_OpeningFcn, ...
                   'gui_OutputFcn',  @GUI_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before GUI is made visible.
function GUI_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to GUI (see VARARGIN)

% Choose default command line output for GUI
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes GUI wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = GUI_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as a
double

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
clc
tune = 1000000*str2num (get (handles.edit5,'String') )       %MHz   %Local
OSC frequency
set_param('finalmodel/tune','Value',num2str(tune/43e6) )
```

```
function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a
double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as a
double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4 as a
double


% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
clc

Fm = 1000*str2num(get(handles.edit2,'String') ) ;           %KHz   %Massege
Freq
Fc = 1000000*str2num (get (handles.edit3,'String') );       %MHz   %Carrier
Freq
Fdiv = 1000*str2num (get (handles.edit4,'String') );       %KHz
%Frequency deviation

set_param('finalmodel/Sine Wave','Frequency',num2str(Fm) );
set_param('finalmodel/FM Modulator Passband','Fc',num2str(Fc) ,
'Kc',num2str(Fdiv));


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

clc
selec = get(handles.no1,'Value');         %Read radiobutton status
set_param('finalmodel/selector','Value',num2str(selec) );
```