André Tiago Pereira Almendra Ferreira

Development of a Spectrum Analyzer based on FPGA

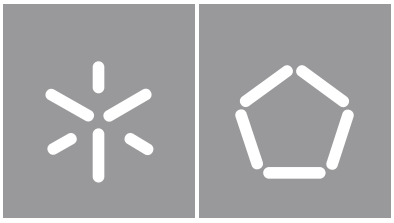
Universidade do Minho
Escola de Engenharia

Janeiro de 2023

Development of a Spectrum Analyzer based on FPGA

André Almendra

UMinho | 2023



Universidade do Minho
Escola de Engenharia

André Tiago Pereira Almendra Ferreira

Development of a Spectrum Analyzer based on FPGA

Dissertação de Mestrado
Engenharia Eletrónica Industrial e Computadores
Sistemas Embebidos e Computadores

Trabalho efetuado sob a orientação de
Professor Doutor Sérgio Lopes
Professor Doutor Sérgio Monteiro

Janeiro de 2023

DIREITOS DE AUTOR E CONDIÇÕES DE UTILIZAÇÃO DO TRABALHO POR TERCEIROS

Acknowledgements

A great deal of knowledge was gained throughout the research and writing of my dissertation, and I would like to use this opportunity to convey my heartfelt appreciation to everyone who has provided vital support and help. This dissertation was supported and developed at Bosch Car Multimedia.

This dissertation would not have been feasible without the support of a number of people. I'd like to thank my supervisor, Dr. Ricardo Gonçalves, for his help, timely and valuable advise, and willingness to share all of his expertise. His advice was crucial in the construction of this dissertation. In terms of knowledge sharing, I'd want to thank every member of the Hardware team for doing so, as well as for their availability to provide help. Furthermore, I want to express my heartfelt gratitude to the Hardware Team Leader for giving me the opportunity to develop my dissertation at Bosch Car Multimedia and to start my career as a Hardware Developer with one of the most wonderful teams I've ever encountered. This was one of the most significant opportunities I've ever had.

I would like to convey my sincere gratitude to my supervisors, Professor Sérgio Lopes and Professor Sérgio Monteiro, for their unwavering support throughout the writing and development of this dissertation. Their assistance, experience and knowledge were crucial for achieving all of the objectives.

Last but not least, I am grateful to my entire friends and family, specially my parents and wife, whose unconditional love, care, and patience made the difficult task of completing this dissertation worthwhile. All of the difficulties would be tough to conquer without their assistance.

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

Resumo

A electrónica no setor automóvel evoluiu imenso durante as duas últimas décadas. A evolução destes sistemas obrigou ao desenvolvimento dos sistemas de comunicação dos veículos, a fim de manter tudo sincronizado, e apesar de terem uma maior eficiência, consomem mais energia, o que significa que as fontes de energia destes sistemas terão que ser mais sofisticadas devido ao número elevado de frequências de funcionamento. Além disso, cada dispositivo eletrónico contém numerosas unidades de processamento que operam em frequências diferentes, emparelhadas com várias interfaces de comunicação. Devido a isto, houve a necessidade de criar regras de conformidade electromagnética (CEM), que impõem testes extensivos para cada dispositivo electrónico, o que fez também aprimorar as regras de conformidade electromagnetica a que estes dispositivos são sujeitos e torná-las cada vez mais restritivas. Estas regras estão centradas tanto nos sinais emitidos pelo dispositivo como na sua capacidade de resistir a interferências.

Para responder à necessidade de um equipamento mais especializado para a resolução de problemas e testes de conformidade, foi utilizada uma Field Programmable Gate Array (FPGA) e um Analog to Digital Converter (ADC) de baixo custo para desenvolver um dispositivo capaz de captar sinais e calcular o seu espectro. Foi utilizada uma arquitectura heteródina para obter uma melhor resolução através da partição do sinal adquirido a partir do ADC. Esta arquitetura é uma técnica de processamento digital de sinal bastante utilizada actualmente, onde são utilizados um oscilador local, um misturador e um filtro passa-baixo. Após segmentar o sinal, é calculado o espectro do segmento do sinal em causa através do algoritmo Fast Fourier Transform (FFT) e a sua potência através do algoritmo Coordinate Rotation Digital Computer (CORDIC).

Com esta arquitectura foi possível reduzir a utilização de recursos do sistema, no entanto, devido a limitações na memória *Block* RAM (BRAM) da FPGA, não foi possível obter um baixo nível de ruído de fundo. Apesar desta limitação, o dispositivo desenvolvido pode ser utilizado para identificar frequências de funcionamento que causem problemas de não conformidade. Este instrumento é essencial para a deteção de sinais na gama de testes de emissões conduzidas, particularmente na gama de Very High Frequency (VHF), onde os sinais de baixa potência podem ser difíceis de distinguir do ruído.

palavras-chave: conformidade eletromagnética, espetro, FFT, FPGA, heteródina, ruído

Abstract

Electronics in the automotive sector have advanced drastically during the last two decades. The evolution of these systems has required the development of vehicle communication systems in order to keep everything synchronized, and although they are more efficient, they consume more energy, which means that the power sources for these systems will have to be more sophisticated due to the high number of operating frequencies. In addition, each electronic device contains numerous processing units operating at different frequencies, paired with various communication interfaces. Because of this, the Eletromagnetic Compliance (EMC) guiding standards have also evolved accordingly and imposed more restrictions on devices, expanded test bands and imposed newer test scenarios. These rules focus on both the signals emitted by the device and its ability to resist interference.

To address the need for more specialized equipment for troubleshooting and compliance testing, a low-cost Field Programmable Gate Array (FPGA) and Analog to Digital Converter (ADC) were used to develop a device capable of capturing signals and calculating their spectrum. An heterodyne architecture was employed to achieve better resolution by partitioning the acquired signal from the ADC. This architecture is a widely used digital signal processing technique today, where a local oscillator, a mixer and a low-pass filter are used. After partitioning the signal, the spectrum of the signal segment in question is calculated using the Fast Fourier Transform (FFT) algorithm and its power using the Coordinate Rotation Digital Computer (CORDIC) algorithm.

With this architecture it was possible to reduce the resource usage of the system, however, due to limitations in the FPGA's block RAM (BRAM), it was not possible to achieve a low noise-floor. Despite this limitation, the developed device is aimed at EMC troubleshooting and can be used to identify operating frequencies that cause non-compliance problems. This instrument is essential for signal detection in the range of conducted emissions, particularly in the Very High Frequency (VHF) range, where low power signals can be difficult to distinguish from noise.

keywords: eletromagnetic compliance, spectrum, FFT, FPGA, heterodyne, noise

Contents

List of Figures

List of Tables

List of Abbreviations

ADC Analog to Digital Converter.

API Application Programming Interface.

ARM Advanced RISC Machines.

AXI Advanced eXtensible Interface.

CAN Controller Area Network.

CAN FD Controller Area Network Flexible Data-Rate.

CISPR Comité International Spécial des Perturbations Radioélectriques.

CORDIC Coordinate Rotation Digital Computer.

DC Direct Current.

DDS Direct Digital Synthesizers.

DFT Discrete Fourier Transform.

DMA Direct Memory Access.

DSP Digital Signal Processing.

DUT Device Under Test.

EDA Eletronic Design Automation.

EMC Eletromagnetic Compliance.

EMI Eletromagnetic Interference.

FFT Fast Fourier Transform.

FIFO First In First Out.

FIR Finite Impulse Response.

FM Frequency Modulated.

FMC FPGA Mezzanine Card.

FPGA Field Programmable Gate Array.

GUI Graphics User Interface.

HDL Hardware Description Language.

HLS High-Level Synthesis.

IDE Integrated Development Environment.

IDFT Inverse Discrete Fourier Transform.

IF Intermediate Frequency.

ILA Integrated Logic Analyzer.

IP Intellectual Property.

LIN Local Interconnect Network.

LSB Least Significant Bit.

LVDS Low-voltage differential signaling.

MM Memory-Map.

MSPS Megasamples per Second.

MUX Multiplexer.

NCO Numeric Controlled Oscillators.

NFFT Nonequispaced Fast Fourier Transform.

PL Processing Logic.

PRBS Pseudo Random Bit Sequence.

PS Processing System.

PSoC Programmable System on a Chip.

RBW Resolution Bandwidth.

RF Radio Frequency.

RMS Root Mean Square.

SDK Software Development Kit.

SDSoC Software-Defined System On Chip.

SoC System on a Chip.

SPI Serial Peripheral Interface.

UART Universal Asynchronous Receiver-Transmitter.

UHF Ultra High Frequency.

VBW Video Bandwidth.

VCO Voltage-Controlled Oscillator.

VHF Very High Frequency.

Vpp Peak-to-Peak Voltage.

Chapter 1: Introduction

1.1 Contextualization

As technology evolves and devices become increasingly more complex, measurement instrumentation has to keep up with the requirements and so these evolve as well to provide more advanced measuring capabilities and with higher specifications. Spectrum analyzers are important for the development and diagnostics of many electronics devices, in Industries ranging from commercial end-user devices, automotive, aerospace and military. They are used to check on the frequency content of signals generated by electronic devices, both intentional and non-intentional signals.

The purpose of this dissertation is to develop such a spectrum analyzer, specially targeted at receiving and displaying the frequency spectrum of signals in the low frequency range. The device should be capable of capturing signals in frequency domain and display the power level in between 9 kHz and 108 MHz with a resolution of 10 Hz. For that purpose, a multi-channel high speed Analog to Digital Converter (ADC) front end should be used in combination with a Field Programmable Gate Array (FPGA) for digital signal processing of the received signals and interface communication with a computer.

1.2 Motivation

In the last two decades, there has been an incredible advance in electronics in the automotive industry. Not only vehicles are being fitted with more electronic devices, but these devices are much more advanced and complex at their core. The increase in electronic systems, leads to an increase in communication systems within the vehicles to have all these things working together in sync. These electronic systems, even though more efficient, are more power hungry because their leap in technology is tremendous with each generation. That means an increase in complexity and operating frequencies in the power supply design of such systems, plus, there are multiple processing units within each device working at different frequencies. All of these together, combined with increased communication interfaces, such as Controller Area Network (CAN), Controller Area Network Flexible Data-Rate (CAN FD), Local Interconnect Network (LIN), Ethernet, and several wireless technologies, means these new generation electronic systems have generators of signals with tremendous harmonic content, which means that they can cause a

lot of interference with each other.

Enters the legal aspect of product development. The legislation concerning the compliance of electronic products has been tightening up on Eletromagnetic Compliance (EMC), and the spectral limits for radiation and immunity are becoming more demanding with each update. Electromagnetic compliance standards require a considerable amount of tests to each electronic device. They focus on both, the signals being radiated by the device, as well as its capability to withstand interference. In both cases, there are several tests and setups that need to be checked, for instance, there is a setup to check radiation of the device in the form of electric/magnetic field, as far-field source as well as near-field source. There are tests to verify its radiation from the device itself and the cable harness associated to the device, and also tests for the unintended signals the device leaks to the power supply cables. An essential tool used for these measurements is a spectrum analyzer. Most spectrum analyzers are very capable devices, although very expensive. For compliance with norms check, there are several commercial solutions available, but there is a time when, for troubleshooting purposes, a more tailored equipment can come in handy. And that is the main driver for this dissertation.

As mentioned earlier, power supplies are becoming more efficient in supplying massive amounts of power while maintaining its operation relatively cool and at high efficiency levels through the whole load variation. These have come at the cost of increasing frequency and variable duty-cycle times. For efficiency this is great, for EMC this is disastrous. But the most tricky thing comes from the fact that many high frequencies, in the Very High Frequency (VHF) and Ultra High Frequency (UHF) range, that causes non-compliance issues in radiated emissions, are actually caused by lower frequency signals that get their harmonics mixed up with communication signals. This can be very hard to spot, and also to troubleshoot the low frequency signal, especially when the low frequency signals are well within EMC limits and many times hidden within the noise floor of the measurements. For that reason, a speciality tool is required to make spectrum analysis in the conducted emissions tests range, which is from a few kHz to a hundred MHz, with very high frequency resolution, with the minimal measurement noise floor possible, to discern very low power signals in this range.

1.3 Objective

The spectrum analyzer to develop should be capable of characterizing the signal's power level between 9 kHz and 108 MHz frequencies, with a resolution of 10 Hz, and provide an interface to access the data from a computer. For that purpose, a multi-channel high speed ADC front end should be used in

combination with an Field Programmable Gate Array (FPGA) for digital signal processing of the received signals and interface communication with a computer. The following objectives are going to be pursued:

- **State of the art screening of acquisition front-ends in frequency for applications with very low noise floor requirements**

  The device is intended for applications such as EMC compliance measures, conducted at low frequencies. For that goal, this device needs to have very low noise floor in order to display signals with very small spectral density in its measurement range.

- **System based on FPGAs and PSoCs**

  The design of the architecture and implementation of the integrated system is based on FPGAs for signal processing and communication interfaces along with high-speed ADC front-end.

- **Creation of communication interface with computer for data presentation**

  A communication interface with computer will be implemented for data presentation, management and storage.

- **Integration and system testing in a measurement environment with real Device Under Test (DUT)**

  Lastly, the system must be tested in a measurement environment with real DUT.

1.4 Dissertation structure

After this chapter, which contains a brief introduction with the context, motivation and project's objective, the chapter 2 has the current state of the art, containing techniques of data acquisition on section 2.1 and various types and arquitectures of spectrum analyzers on section 2.2, as well as its operating principle. This is followed by chapter 3, where the components regarding analysis and design of the system are described. On section 3.1 the hardware and software resources are described, as well as a brief description of the IP Cores required to build the spectrum analyzer and the communication protocol used between the PL and the PS. Section 3.3 describes the ADC clock source and its setup possibilities. On section 3.4 it is described the system architecture used, explains why this architecture was used and its pros and cons. Section 3.5, the final section of this chapter, depicts the use-case for the utilization of the spectrum analyzer.

Chapter 4 explains the development of the low noise spectrum analyzer. This chapter is divided into three sections. On Section 4.1, the main IP Cores from Analog Devices reference design are described, as well as the used communication protocol, followed by Section 4.2, composed by all the simulations made to the system. Section 4.3 has all the steps taken in the system implementation, starting with spectrum calculation in 4.3.1, followed by an analysis and comparison of the obtained results, system requirements, limitations. 4.3.3 and 4.3.4 shows the final implementation, its test results and the developed user interface for the spectrum analyzer with test results for each step of the implementation.

In Chapter 5, the spectrum analyzer's noise floor is analyzed, along with DUT tests performed for this end, along with possible solutions to optimize the spectrum analyzer's characteristics and limitations.

Lastly, Chapter 6 provides a summary and highlights important considerations regarding this dissertation, along with other possible solutions that can make this spectrum analyzer meet its requirements.

Chapter 2: Background and State of the Art

This chapter describes high-speed FPGA-based data acquisition systems, spectrum analyzers, their architecture and its different topologies, starting with data acquisition systems using the high speed processing feature of FPGAs, an overview of these systems and its components, followed by an explanation of why nowadays, devices such as FPGAs are used on these kinds of systems. This is followed by an overview of spectrum analyzers, namely their architectures, topologies and functions. All of these subjects are key components in implementing a spectrum analyzer using an FPGA.

2.1 Data acquisition

The physical phenomena are continuous in nature and can be measured in the analog domain, whereas the digital domain consists of magnitudes that are measured in discrete-time. The translation of these constraints from analog to digital signal domain is the key enhancement in data acquisition systems. A data acquisition system is a method of sampling signals that measure physical quantities and convert the result into digital values that can be manipulated and saved for further use, and it is currently the most important component in every measurement system. For example, it is used in power quality monitoring [10], spectroscopy devices [11], in the characterization of energy-harvesting devices [12], in real-time reflectometry diagnostic [13], in fusion experiments [14], in power metering and many other applications. This is necessary to detect, acquire, process and analyse data. A data acquisition system is often built around a host computer that communicates with Analog to Digital Converter (ADC) and the host computer's primary role is to handle graphical presentation or do tiny computations, mostly offline, or if online, with timings dependent on the event rate, although in many situations not negligible in comparison to the ADC's conversion time [15]. The introduction of ADCs was the cornerstone of a great change in measuring systems and device topologies. The rapid development of fast ADCs capable of attaining resolutions greater than 10-12 bit has transformed the instrumentation and measurement fields and may be regarded as one of the critical building elements of our global digital content-driven civilization [16]. The ability to measure almost anything, sometimes at extremely low cost, has multiplied the deployment of sensor-based measurement systems as well as the total amount of acquired data, the rapid development of signal processing algorithms to extract real-time information from raw data, and the use of data min-

ing to analyze the results. A data acquisition system is a system or device that can digitize an electrical quantity at a specific sampling rate. It might have numerous input channels with different input ranges or just one. By far the majority of commercial systems are intended to obtain voltages. For multi-channel devices, the simplest option is to use a single ADC and sequentially link the input signals to the ADC input (through a multiplexer), allowing the acquisition of several channels with a single ADC [16].

2.1.1 DSP-based data acquisition

A traditional data acquisition system usually uses a Digital Signal Processing (DSP), however, using a single chip or a DSP as a control device has the disadvantage of low processing speed and simple functions. Even though the use of a DSP can perform a high-speed data acquisition, the cost increases as the processing speed goes up. Also, a traditional system consists of a large number of discrete devices and complex circuits. The reliability of a traditional data acquisition system is not high and it is difficult to debug.

2.1.2 FPGA-based data acquisition

Nowadays, FPGAs have specific applications in data acquisition due to its features, such as soft and hard core processors, logic gates and dedicated multipliers. With its high processing speed and reconfiguration level, the problems mentioned in the DSP-based data acquisition systems can be solved. An FPGA-based data acquisition system is able to convert analog signals produced by various sensors to digital signals and send them to external devices, such as personal computers or external drives.

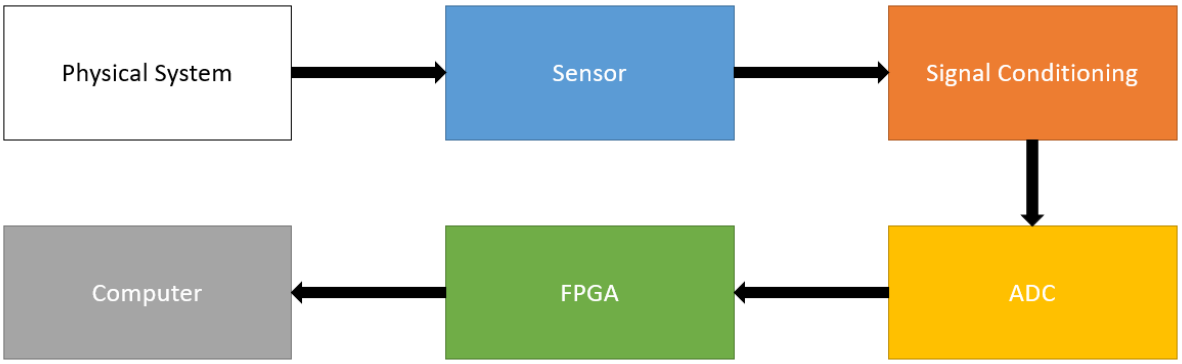


Figure 1: FPGA-based data acquisition system block diagram

FPGA based data acquisition systems are designed to record real time data that comes from interfaces like digital and analog channels, serial channels and CAN Bus. The data acquisition system is used as a

compact and modular device which is capable of data conversion, collection and transmission. Design is performed with an FPGA because of its high processing speed feature [17], because although computers have processors, they are generic processors and can't keep up with the processing speed of an FPGA, in which is implemented a dedicated logic to perform direct calculations according to the required functions. This is the reason why top tier modern measuring devices use FPGAs to perform the signal processing before sending data to an external device, such as computers.

2.2 Spectrum analyzer operating principle

A spectrum analyzer, like an oscilloscope, displays on a screen an input signal. The main difference between these two is that an oscilloscope is used to display the waveform of a signal in the time domain and the spectrum analyzer is used to display the input signal in the frequency domain, although it is possible to display the time waveform with proper settings. A spectrum analyzer cannot handle large signals, so before connecting a signal to the input, it must be ensured that the signal will not exceed its maximum allowable input.

2.3 Spectrum analyzer topologies

Spectrum analyzers are classified into two types: swept-tuned analyzers and real-time analyzers. Modern spectrum analyzers employ digital signal processing to give extra measuring capacity and make it much easier to comprehend test findings. Amplitude versus frequency is displayed by both swept-tuned and real-time spectrum analyzers. However, how they process and show this information differs depending on the type of analyzer. A real-time spectrum analyzer is able to display the amplitude of the signal at all frequency components at the same time, as opposed to a swept-tuned spectrum analyzer, which displays measurement data sequentially and does not display data in real time. This is because a swept-tuned analyzer produces a spectrum display by using a single narrow filter tuned across a range of frequencies [2]. There are several architectures for swept-tuned and real-time spectrum analyzers, such as parallel or multi-channel spectrum analyzers, tuned radio frequency spectrum analyzers, superheterodyne spectrum analyzers, Fourier or FFT analyzers, vector signal analyzers, and real-time spectrum analyzers [18]. This section discusses each spectrum analyzer architecture mentioned previously.

2.3.1 Parallel-filter spectrum analyzers

The earliest spectrum analyzer concept resulted from the use of numerous basic band-pass filters in parallel, each set at a different frequency, as depicted in Figure 2.

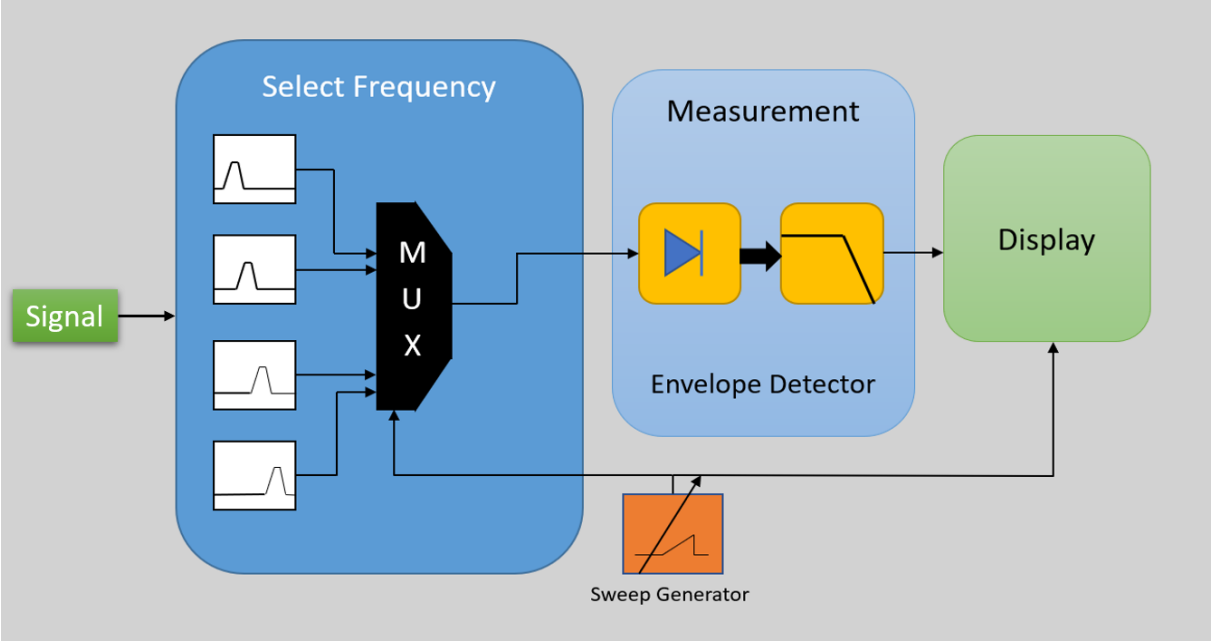


Figure 2: Multi-channel spectrum analyzer block diagram

Following an initial settling time, the parallel-filter analyzer detects and shows all signals within the spectrum analyzer's measuring range in real time. As a result, the parallel-filter analyzer can capture and display signals in real time [2]. Each output is a channel that may be used to calculate amplitudes at the filter's frequency. The advantage of this design is that high resolutions can be achieved by using a large number of filters and the quality of the filters can be great because they are at a fixed frequency, however, filters become hard to make at higher frequencies, as the quality factor of the band-pass filter Q, as shown in Eq.2.1, increases with frequency, with f_c being the frequency in the center point of the band-pass filter and δf_{BW} being the difference between the lower and upper cutoff frequencies.

$$Q = \frac{f_c}{\Delta f_{BW}} \tag{2.1}$$

The parallel-filter analyzer's special strength is its measurement speed, which allows it to detect transient and time-variant signals (also known as dynamic signals), and because they are all in parallel, the speed is quite fast, providing a steady picture. The width of the bandpass filters, on the other hand, determines the frequency resolution. To obtain correct resolution across a wide frequency range, numerous separate filters would be required, increasing the cost and complexity of such an analyzer. This is why,

with the exception of the most basic parallel-filter analyzers, all parallel-filter analyzers are costly. Lastly, channel mismatch reduces amplitude accuracy when utilizing amplitudes from numerous channels [2]. This type of spectrum analyzers was typically used in audio-frequency analyzers.

2.3.2 Swept-tuned or tuned radio frequency spectrum analyzers

The classic design that originally enabled engineers to do frequency domain measurements many decades ago is the swept-tuned, superheterodyne spectrum analyzer. Originally constructed with exclusively analog components, the swept-tuned spectrum analyzer has grown in parallel with the applications it supports. Spectrum analyzers of today incorporate digital components such as ADCs, DSPs and microprocessors. The fundamental sweeping technique, on the other hand, stays substantially unchanged and is best suited for studying controlled, static signals. As shown in Figure 3, the spectrum analyzer measures power versus frequency by downconverting the signal of interest and sweeping it over the passband of an Resolution Bandwidth (RBW) filter.

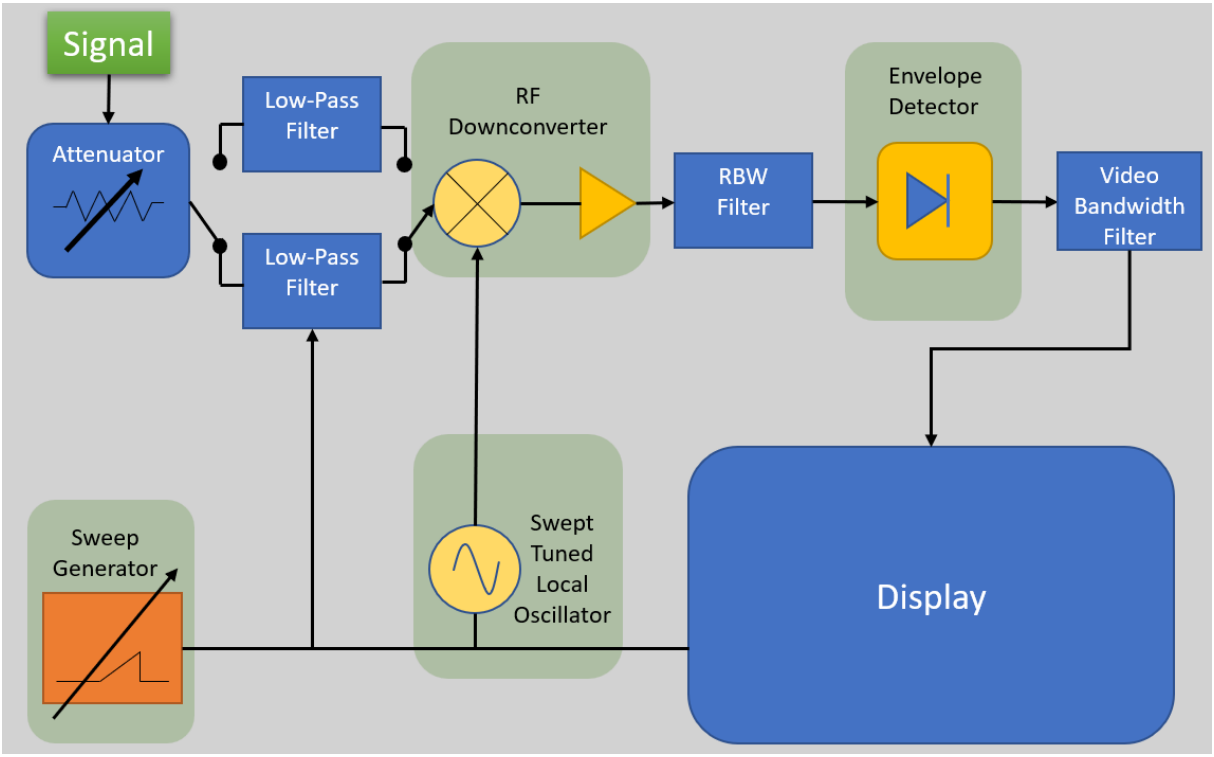


Figure 3: Swept-tuned spectrum analyzer block diagram

Following the RBW filter is a detector that estimates the amplitude at each frequency point in the chosen span. While this approach has a great dynamic range, it has the limitation of only being able to calculate the amplitude data for one frequency point at a time. This method assumes that the analyzer can complete at least one sweep without causing substantial modifications to the signal being monitored.

As a result, measurements are only applicable to highly steady, constant input signals. If the signal varies rapidly, it is probable that some changes will be overlooked. The spectrum analyzer is looking at frequency segment Fa while a transient spectral event happens at Fb, as illustrated in Figure 4 (diagram on left). The event has gone by the time the sweep comes to segment Fb and is not noticed (diagram on right) [1].

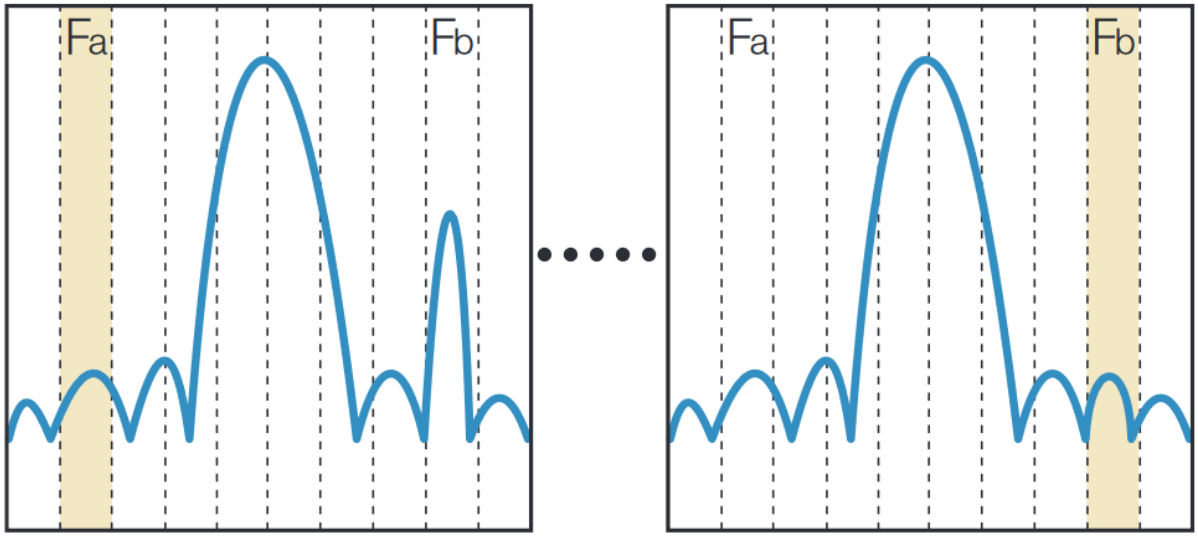


Figure 4: Frequency sweep [1]

This spectrum analyzer architecture does not provide a reliable way to discover the existence of this kind of transient signal, thus contributing to the long time and effort required to troubleshoot many modern Radio Frequency (RF) signals. In addition to missing momentary signals, the spectrum of impulse signals such as those used in modern communications and radar may be misrepresented as well. Spectrum analyzer architectures cannot represent the occupied spectrum of an impulse without repetitive sweeps. One also needs to pay special attention to sweep rate and resolution bandwidth [1].

Even though modern spectrum analyzers have replaced analog functionality with DSP, the fundamental architecture and its limitations remain.

Superheterodyne topology

The superheterodyne method, which is the most often used method and it is used in swept-tune spectrum analyzers, is based on the principle of relative frequency movement between the signal and a filter. The relative frequency movement is the most essential metric. It makes no difference whether the signal is stationary or non-stationary in frequency and the filter is changed. Heterodyne means to mix and super refers to frequencies above the auditory range, these analyzers sweep throughout the frequency range of interest, displaying all of the current frequency components [19].

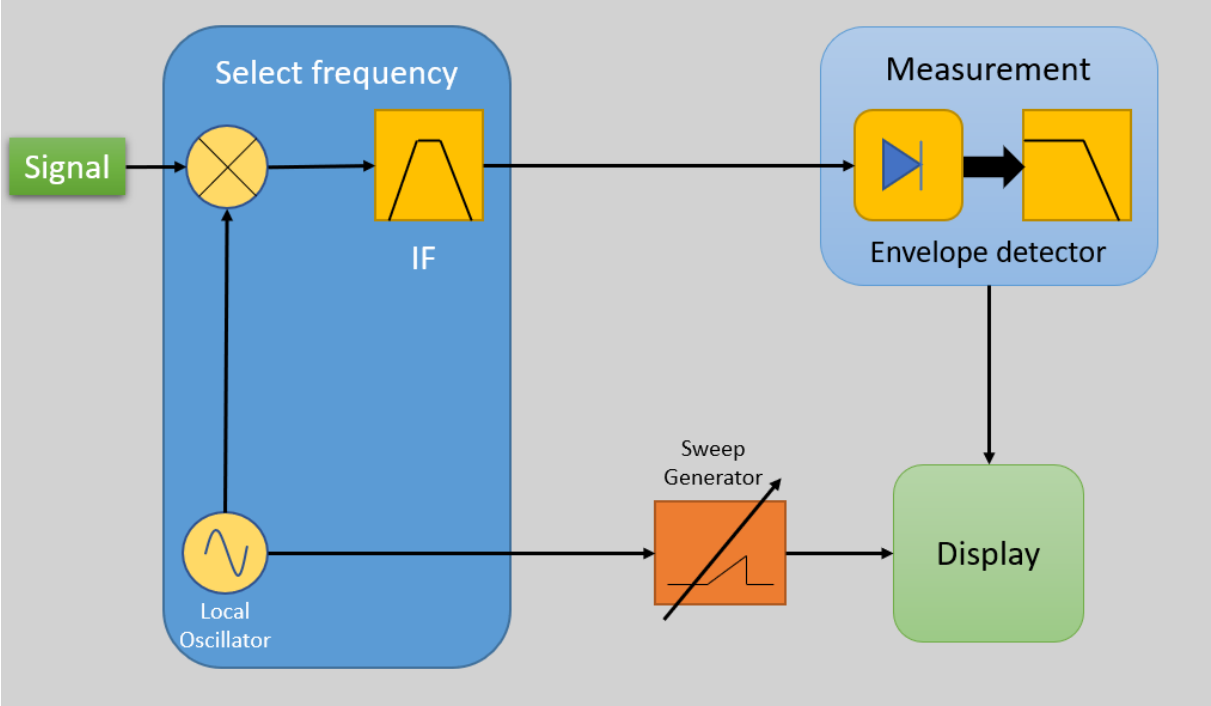


Figure 5: Superheterodyne Spectrum analyzer Block Diagram

The superheterodyne spectrum analyzer, depicted in Figure 5, uses a mixer to transform an input frequency to a preset intermediate frequency (represented as IF). The local oscillator (represented as LO) that mixes with the input signal may then be utilized to adjust which input frequency is translated to the IF. The IF is subsequently band-pass filtered, measured, and displayed.

The fixed frequency band-pass filter and optional amplifier are a benefit of this design. This implies that they can be of high quality, especially if a low IF is used. This also implies that everything after the mixer is independent from the input range. The frequency range of the LO and the mixer, which might be relatively large, determines the input range. The conversion can also be done in phases, allowing for even wider input ranges and improved filtering [18].

This spectrum analyzer architecture is simple and inexpensive, however, there is a disadvantage. The disadvantage is that, depending on the quality of the mixer, distortions are introduced. In general, each level of the design produces more distortions, and this is an additional stage.

This architecture enables frequency domain measurements to be made over a large dynamic range and a wide frequency range, thereby making significant contributions to frequency-domain signal analysis for numerous applications, such as (1)manufacture and maintenance of microwave communications links, radar, telecommunications equipment, cable TV systems, and broadcast equipment, (2)mobile communication systems, (3)EMI diagnostic testing, (4)component testing and (5) signal surveillance [19].

2.3.3 Fourier or FFT spectrum analyzers

An FFT spectrum analyzer directly digitizes the input signal by using an ADC. To optimize the input signal for optimum ADC resolution, an amplifier and/or anti-alias filter may be used. The digitised input signal in the time domain is then further processed digitally and converted to the frequency domain by an FFT [18]. This architecture is shown in Figure 6, followed by an example in Figure 7.

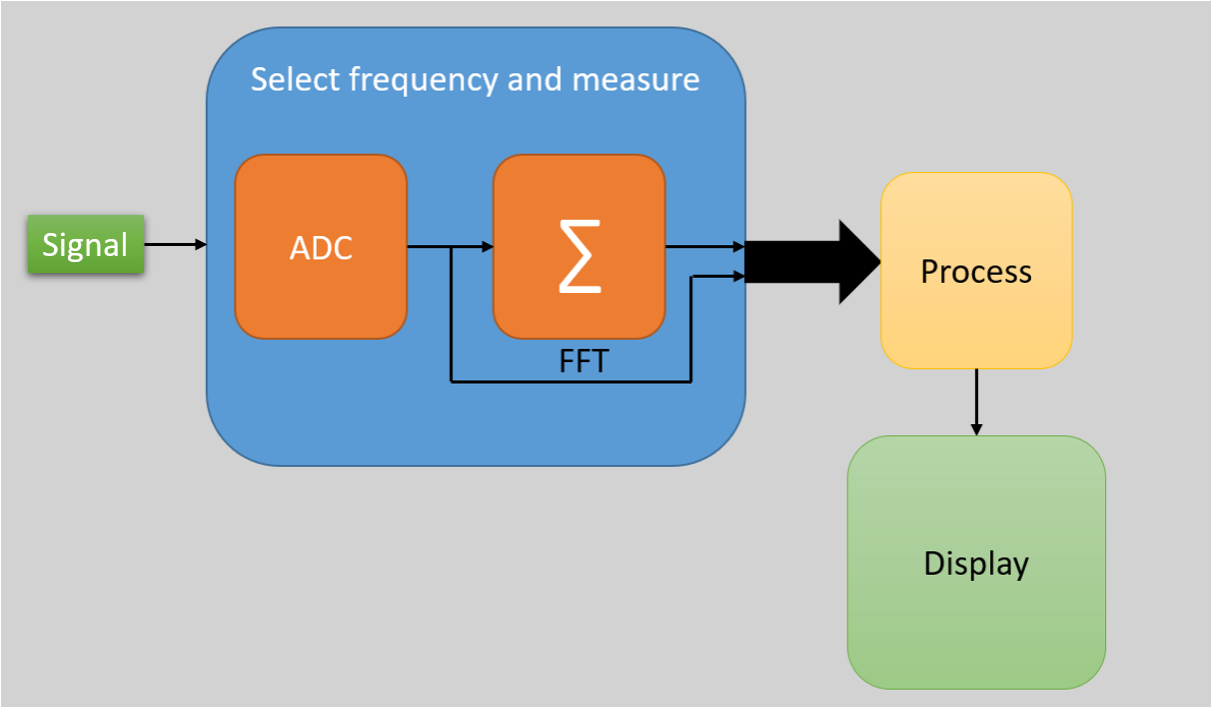


Figure 6: FFT Spectrum analyzer Block Diagram

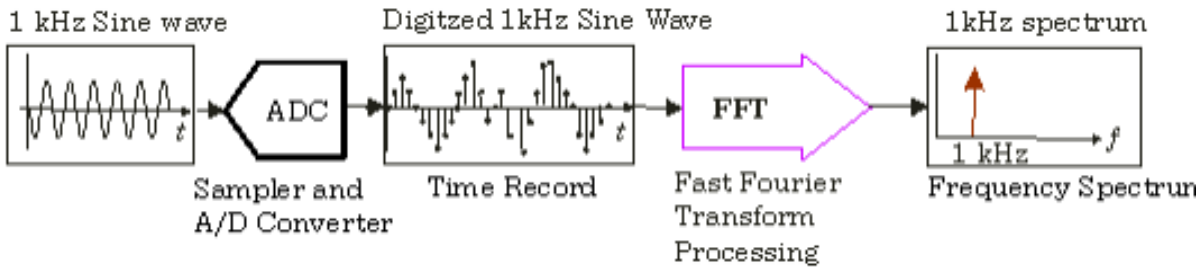


Figure 7: 1 kHz FFT Analysis example - Digitize a time-domain signal and use FFT analysis to convert it to the frequency-domain [2]

FFT spectrum analyzers are powerful tools because they can extract more information from an input signal than just the amplitude of particular frequency components. For example, they can detect both magnitude and phase and can simply move between the time and frequency domains. This makes them useful for analyzing telecommunication signals, ultrasonic signals, and modulated signals [19].

Because a Fourier transform of a sampled input signal is done instead of an average voltage measurement, it is possible to conduct time, frequency, modulation, and coding domain analysis, as well as characterize single-shot events [18]. FFT analyzers formerly had the problem of having a limited frequency range, since most FFT spectrum analyzers could not produce measurements over 100 kHz. The speed of the analog-to-digital converter utilized to sample the analyzer's input signal has been the limiting issue. This is why swept-tuned superheterodyne analyzers are still used for RF and microwave measurements, even though certain newer-generation swept-tuned analyzers can also do FFT measurements [2].

2.3.4 Hybrid FFT spectrum analyzer

The hybrid FFT-based analyzer has similar input architecture to the swept-tuned analyzer, but it digitizes the IF from the mixer using a fast Fourier transform of the input signals. The use of digital filtering and the related increased filter shape parameters and settling time is advantageous. The analyzer will typically gather spectral data, digitize it, run computations, and then show the results. Due to processing time, there may be a pause (or dead time) between taking and showing the spectral data. As depicted in Figure 9, when the computation time exceeds the time record window, the analyzer doesn't operate in real time and causes a dead time, due to the gaps between time records [20].

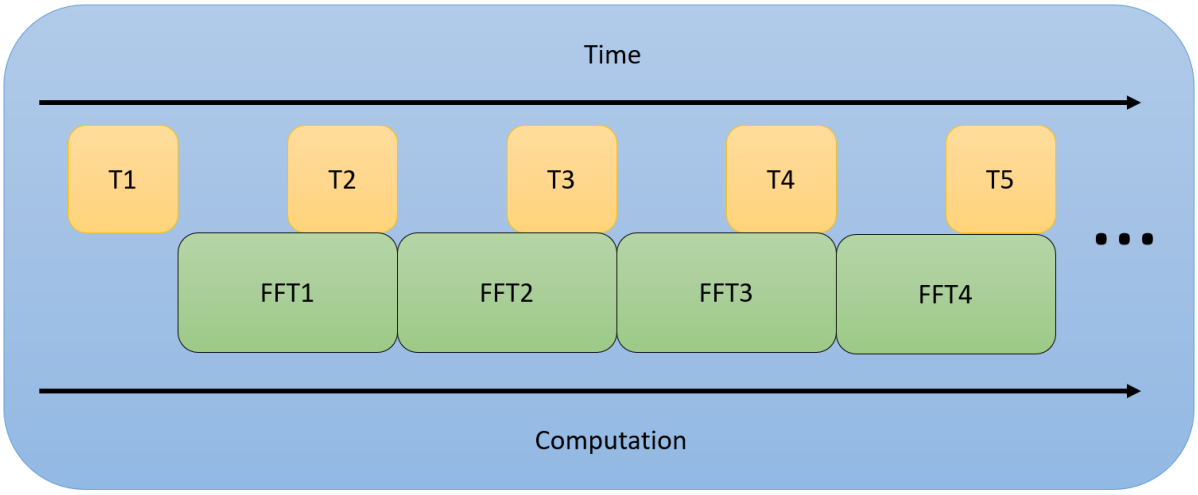


Figure 8: Processing diagram of hybrid FFT spectrum analyzers

However, with today's technological advances in ADCs and FFT processing, the difference between a hybrid FFT analyzer and a real-time analyzer is minimal. Both now use the same basic structure with a fixed LO that is stepped if the span exceeds the FFT window [20].

2.3.5 Vector signal spectrum analyzers

The vector signal spectrum analyzer is similar to the superheterodyne analyzer in architecture, but it digitizes the input signal at the IF and stores the digital waveform in memory [18]. The waveform in memory comprises magnitude and phase information that DSP can utilize for demodulation, measurements, or display processing. An ADC digitizes the wideband IF signal within the VSA, and numerical downconversion, filtering, and detection are done. DFT (discrete Fourier transform) techniques are used to convert from the time domain to the frequency domain. The modulation parameters measured by the VSA are FM deviation, Code Domain Power, and Error Vector Magnitude (EVM and constellation diagrams). Parameters such as channel power, power versus time, and spectrograms are also measured [1]. VSAs use superheterodyne technology in conjunction with high-speed ADCs and other DSP technologies to provide quick, high-resolution spectrum measurements, demodulation, and sophisticated time-domain analysis. The VSA is particularly effective for describing complicated signals such as burst, transient, or modulated signals used in communications, video, broadcast, sonar, and ultrasound imaging [2].

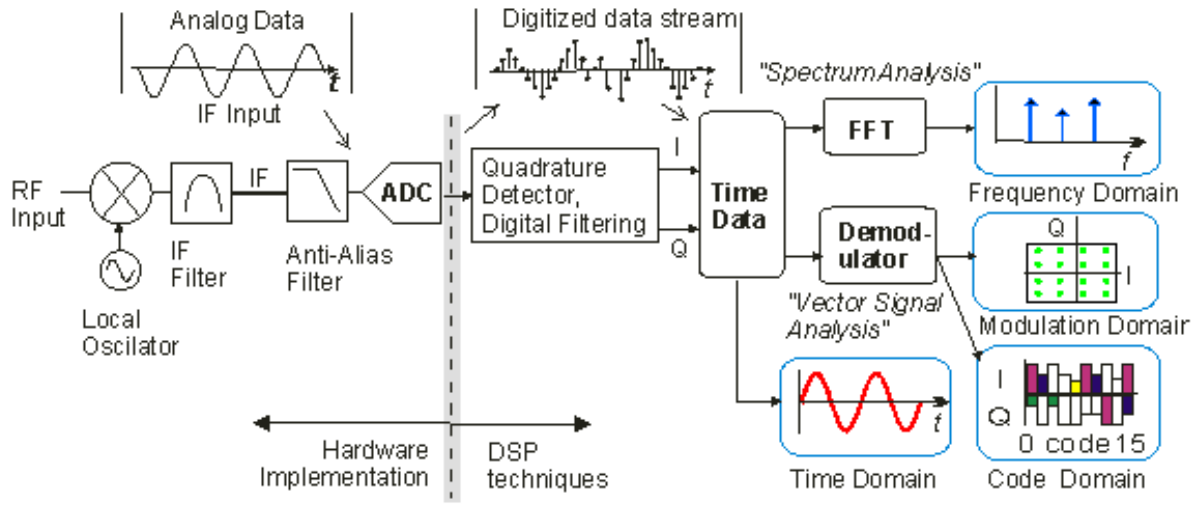


Figure 9: Vector signal spectrum analyzer Block Diagram [2]

While the VSA can now store waveforms in memory, it is still restricted in its capacity to evaluate transitory occurrences. Signals must be saved in memory before being processed in the conventional VSA free run mode. Because the batch processing is serial, the instrument is basically oblivious to events that occur between acquisitions. Single or uncommon incidents cannot be reliably discovered. Triggering on these infrequent occurrences can be used to isolate them in memory. Unfortunately, the triggering capabilities of VSAs are restricted. External triggering necessitates previous awareness of the occurrence in issue, which is not always feasible [1].

2.3.6 Real-time spectrum analyzers

Despite their excellent performance, current superheterodyne analyzers still cannot assess and show a whole frequency spectrum at the same time. As a result, they are not real-time analyzers. Furthermore, measurement periods may be rather long since the sweep speed of a swept-tuned analyzer is always restricted by the time it takes for its internal filters to settle. There are several designs and measurement procedures utilized to deliver real-time, dynamic signal analysis [2].

To evaluate signals in real-time, the analysis processes must be conducted quickly enough to process all signal components in the frequency region of interest properly. This definition indicates that: (1)sampling must be fast enough to fulfill the Nyquist theorem and (2)all calculations must be continuous and fast enough such that the analysis output keeps up with changes in the input signal [1].

While vector signal analyzers may do snap-shot analysis in a variety of domains, their triggering capabilities are restricted, as mentioned before in 2.3.5. However, many observed signals alter unpredictably over time. These devices, sometimes known as real-time analyzers, can detect bursts, glitches, and other time-varying events and collect and interpret the resulting data. This vector signal analyzer extension allows for the capture of critical transient events, however it is largely accomplished in software using the same front-end architecture as the vector signal analyzer [18].

The Real-Time Spectrum Analyzer architecture is intended to overcome the measurement limits of swept-tuned spectrum analyzers and VSAs in order to better meet the issues associated with transient and dynamic RF signals, as discussed in the preceding sections. Modern RTSA execute signal analysis utilizing real-time DSP prior to memory storage, as contrast to the VSA architecture's post-acquisition processing. Real-time processing enables the user to detect events that are undetectable to conventional architectures and to react on those events, allowing selective memory capture. Using batch processing, the data in memory may then be thoroughly evaluated across many domains. In addition, the real-time DSP engine is utilized for signal conditioning, calibration, and some sorts of analysis [1]. An example of a block diagram of a RTSA is depicted in Figure 10.

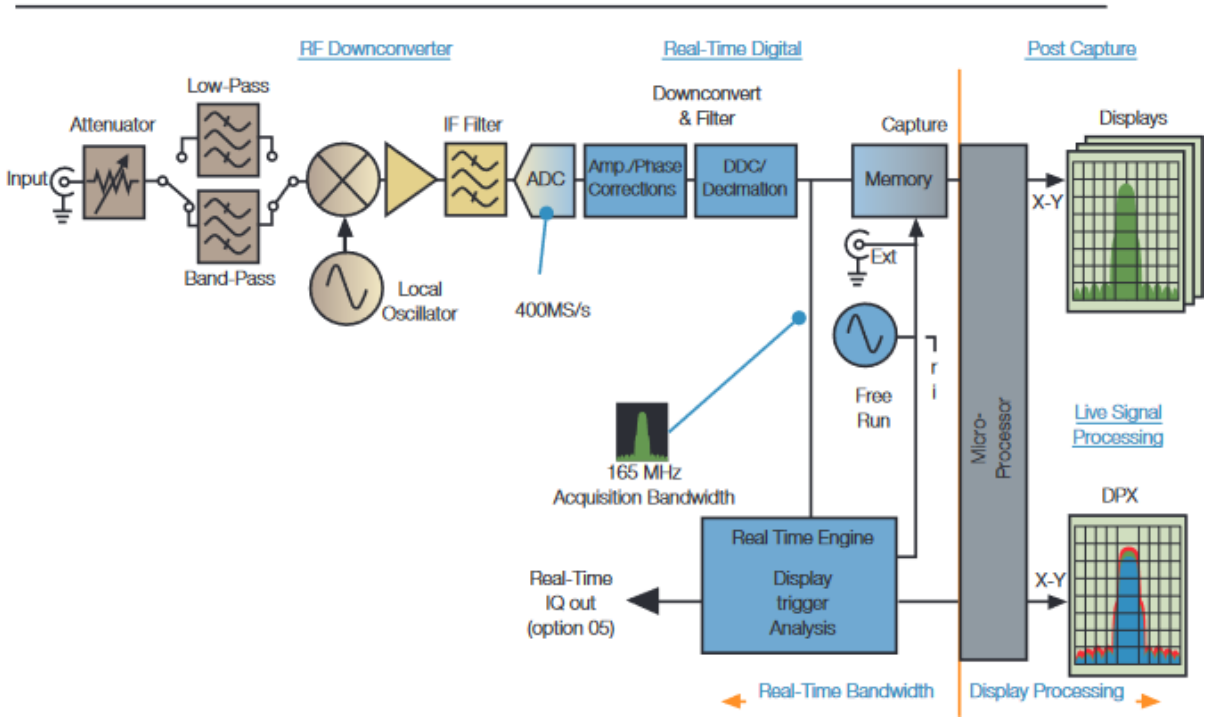


Figure 10: Block diagram of a RTSA (Tektronix RSA5100) [1]

In contrast to the VSA, the real-time engine is fast enough to process every sample without gaps as shown in Figure 11.

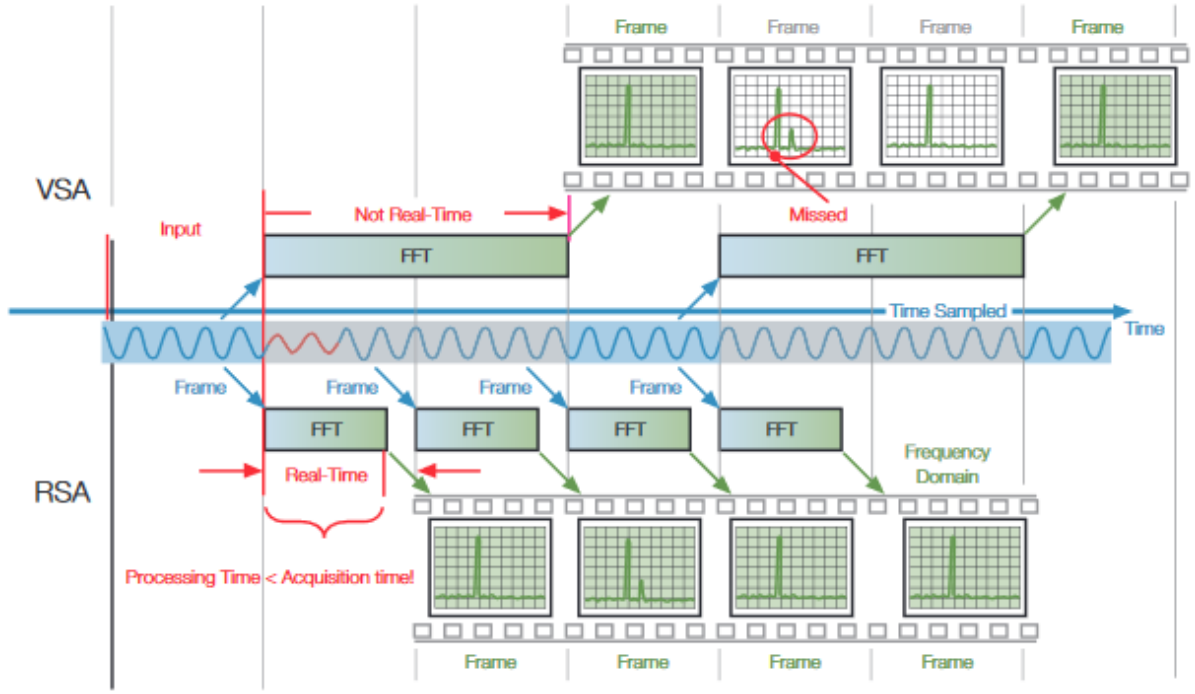


Figure 11: VSA processing vs. Real-Time Spectrum Analyzers real-time engine processing [1]

Continuous amplitude and phase adjustments for analog IF and RF responses are possible. Not only

may the data in memory be entirely repaired, but all future real-time processing can also function on corrected data.

2.4 Fast Fourier Transform

The Fast Fourier Transform started when Carl Friedrich Gauss, a renowned German mathematician, created a method capable of determine the orbit of certain asteroids from sample locations [21]. This led to the development of the Discrete Fourier Transform (DFT).

The study of trigonometric series may be traced back to the work of mathematician Leonhard Euler, although at the time, he had no specific use in mind. He worked with infinite, cosine-only series and was unconcerned with convergence.

Alexis-Claude Clairaut, a French mathematician contemporaneous with Euler and aware of his study, published what is now thought to be the first formula for the DFT in 1754, but it was limited to a cosine-only finite Fourier series. Later, Joseph Louis Lagrange developed a sine-only series DFT-like formula. Clairaut and Lagrange were interested in orbital mechanics and the issue of knowing the specifics of an orbit based on a limited number of observations. As a result, their data was periodic, and they determined orbits using an interpolation method. In current terminology and notation, an even periodic function $f(x)$ with a period of one is expressed as a finite trigonometric series, as shown in equation 2.2.

$$f(x) = \sum_{k=0}^{N-1} a_k cos(2\pi kx), \qquad 0 < x \leq 1. \tag{2.2}$$

Gauss expanded this work on trigonometric interpolation to periodic functions that aren't necessarily odd or even. This was done while thinking about the difficulty of estimating the orbit of certain asteroids based on sample locations. Fourier series is used to represent these functions, as indicated in equation 2.3.

$$f(x) = \sum_{k=0}^{m} a_k cos(2\pi kx) + \sum_{k=1}^{m} b_k sin(2\pi kx) \tag{2.3}$$

In this series, $m = (N-1)/2$ for N odd and $m = N/2$ for N even. Gauss demonstrated in his interpolation paper that if the values of $f(x_n)$, $X_n = n/N$, where $n = 0, 1, ..., N-1$ are supplied, the coefficients a_k and b_k are obtained by the now well-known DFT formulae. This collection of equations is the first explicit formula for generic DFT discovered. Gauss created his efficient approach by employing N evenly spaced samples across one period of the signal. During the development of the DFT

method, Gauss discovered a condition known as undersampling, which produced errors in the coefficients due to aliasing of the high-frequency harmonics. His approach was to measure a total of N_2 sets of N, equally spaced samples, resulting in an overall set of $N = N_1 N_2$ equally spaced samples. Equation 2.4 represents the DFT of $f(x)$ samples, where $X(n) = f(n/N)$ represents the N evenly spaced samples, $W_N = e^{-j2\pi/N}$, and $k = 0, 1, ..., N-1$ are the Fourier coefficient indices.

$$f(x) = \sum_{n=0}^{N-1} X(n) W_N^{nk} \tag{2.4}$$

This DFT can be rewritten in terms of N_2 sets of N subsamples by the change of index variables, as depicted in Equations 2.5 and 2.6, where for n_1, $k = 0, 1, ..., N_1 - 1$ and n_2, $k_2 = 0, 1, ..., N_2 - 1$.

$$n = N_2 n_1 + n_2 \tag{2.5}$$

$$n = k_1 + N_1 k_2 \tag{2.6}$$

Each subsequence is a function of n_1 and which subsequence it is, is denoted by n_2. As a result, the DFT becomes the equation depicted in 2.7.

$$C(k_1 + N_1 k_2) = \sum_{n_2=0}^{N_2-1} \left[\sum_{n_1=0}^{N_1-1} X(N_2 n_1 + n_2) W_{N_1}^{n_1 k_1} W_N^{n_2 k_2} \right] W_{N_2}^{n_2 k_2}. \tag{2.7}$$

The inner sum calculates the N_2 length$-N_1$ DFT, which is corrected by a power of W_N, while the outer sum computes the N_1, length$-N_2$ DFT. This is the exponential form of Gauss' method, with the W_N term accounting for shifts from the origin of the N_2 length$-N_1$ sequences.

J.W.Cooley and J.W.Tukey's independently re-invented this algorithm, where W_N is designated the twiddle factor, trigonometric constant coefficients that are multiplied by the data in the course of the algorithm [22]. They released a study on the FFT algorithm as a way of calculating the DFT. This was a groundbreaking point in digital signal processing and several numerical analysis areas. They also showed that the DFT, which was previously thought to need N^2 arithmetic operations, could be calculated with a number of operations proportional to $Nlog_i(N)$ using the new FFT technique. This technology revolutionized most of digital processing, and the Fourier transform is still the most widely used method of executing it [21].

2.4.1 Fast Fourier Transform implementations on FPGA

J.W.Cooley and J.W.Tukey pioneered the concept of FFT by reducing computation complexity through the symmetry and periodicity aspects of the twiddle factors [22]. The FFT and inverse FFT algorithms are key processing blocks for translating data from the time-to-frequency domain (FFT) and frequency-to-time domain (IFFT). The FFT processor's efficacy is critical in maximizing system performance.

The two types of Cooley-Tukey FFT algorithms are the decimation in time (DIT) and decimation in frequency (DIF). These two kinds are distinguished by the order of input and output samples. The simplest and most typical variant of the Cooley-Tukey method is the Radix-2.

Radix-2

Radix-2, as the name implies, splits a DFT of size N into two interleaved DFTs of size $N/2$ at each recursive stage. If $N = 2^v$, then $v = log_2 N$. In comparison to the order of N^2 in DFT, the number of complex multiplications and additions are now decreased to $N/2log_2 N$ and $Nlog_2 N$, respectively [3].

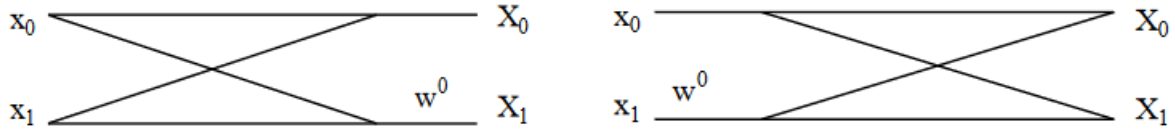


Figure 12: Radix-2 butterflies, (a) DIF (b) DIT [3]

Radix-2 butterflies for DIF and DIT are shown in Figure 12

RADIX-4

The Radix-4 FFT algorithm divides the discrete Fourier transform (DFT) equation's N-point input sequence into four subsequences, as depicted in 2.8.

$$(x(4n), x(4n+1), x(4n+2), x(4n+3), n = 0, 1, ..., N/4-1) \tag{2.8}$$

Figure 13 depicts Radix-4 butterflies for DIF and DIT. It should be noted that $W_N^0 = 1$ requires three complex multiplications and twelve complex additions for each butterfly. If $v = 4$ for Radix-4, the number of points $N = 4^v$. FFT method consists of v phases, each of $N/4$ butterflies. As a result, $3vN/4 = (3N/8)log_2 N$ complex multiplications and $(3N/2)log_2 N$ complex additions are necessary [3]. In comparison to Radix-2, there is a 25% reduction in multiplications, but only 50% of additions are necessary in Radix-2 [23].

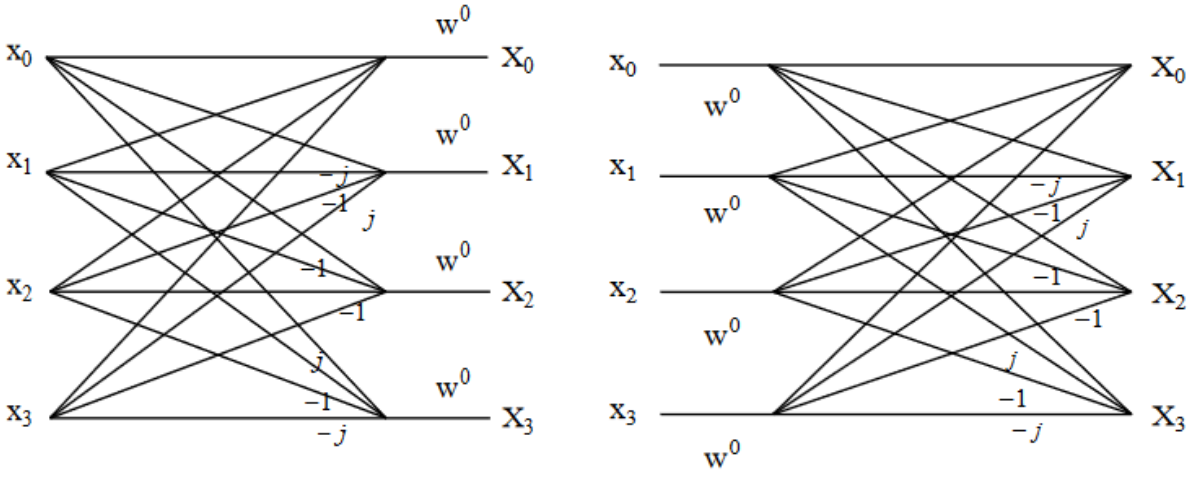


Figure 13: Radix-4 butterflies, (a) DIF (b) DIT [3]

When compared to Radix-2 algorithm, Radix-4 algorithm has the benefit of requiring fewer complex multiplications, but it requires 3N complex additions.

Xilinx currently provides FFT IP Cores with multiple configurations such as N-points and FFT implementations such as Radix-2 and Radix-4.

2.5 CORDIC algorithm

The Coordinate Rotation Digital Computer (CORDIC) algorithm offers an opportunity to calculate trigonometric functions, coordinate transformations, vector rotations, or hyperbolic rotations in a rather simple and elegant way. The CORDIC algorithm was first introduced by Jack E. Volder, hence the CORDIC algorithm being known also as Volder's algorithm. John Stephen Walther later developed it into a unified algorithm to compute a variety of transcendental functions. Two basic CORDIC modes leading to the computation functions are the rotation mode and the vectoring mode. For both modes, the algorithm can be used as an iterative sequence of additions, subtractions and shift operations which are rotations by a fixed rotation angle, but with a variable rotation direction.

The Coordinate Rotation Digital Computer (CORDIC) algorithm makes it possible to calculate trigonometric functions, coordinate transformations, vector rotations, and hyperbolic rotations in a straightforward and elegant manner. The CORDIC algorithm was invented by Jack E. Volder, which is why it is also known as Volder's algorithm. Later, John Stephen Walther expanded it into a unified algorithm for computing a variety of transcendental functions [24]. The rotation mode and the vectoring mode are two fundamental CORDIC modes that lead to computation functions. The technique may be used in both modes as an iterative series of adds, subtractions, and shift operations that are rotations with a constant rotation angle

but a variable rotation direction.

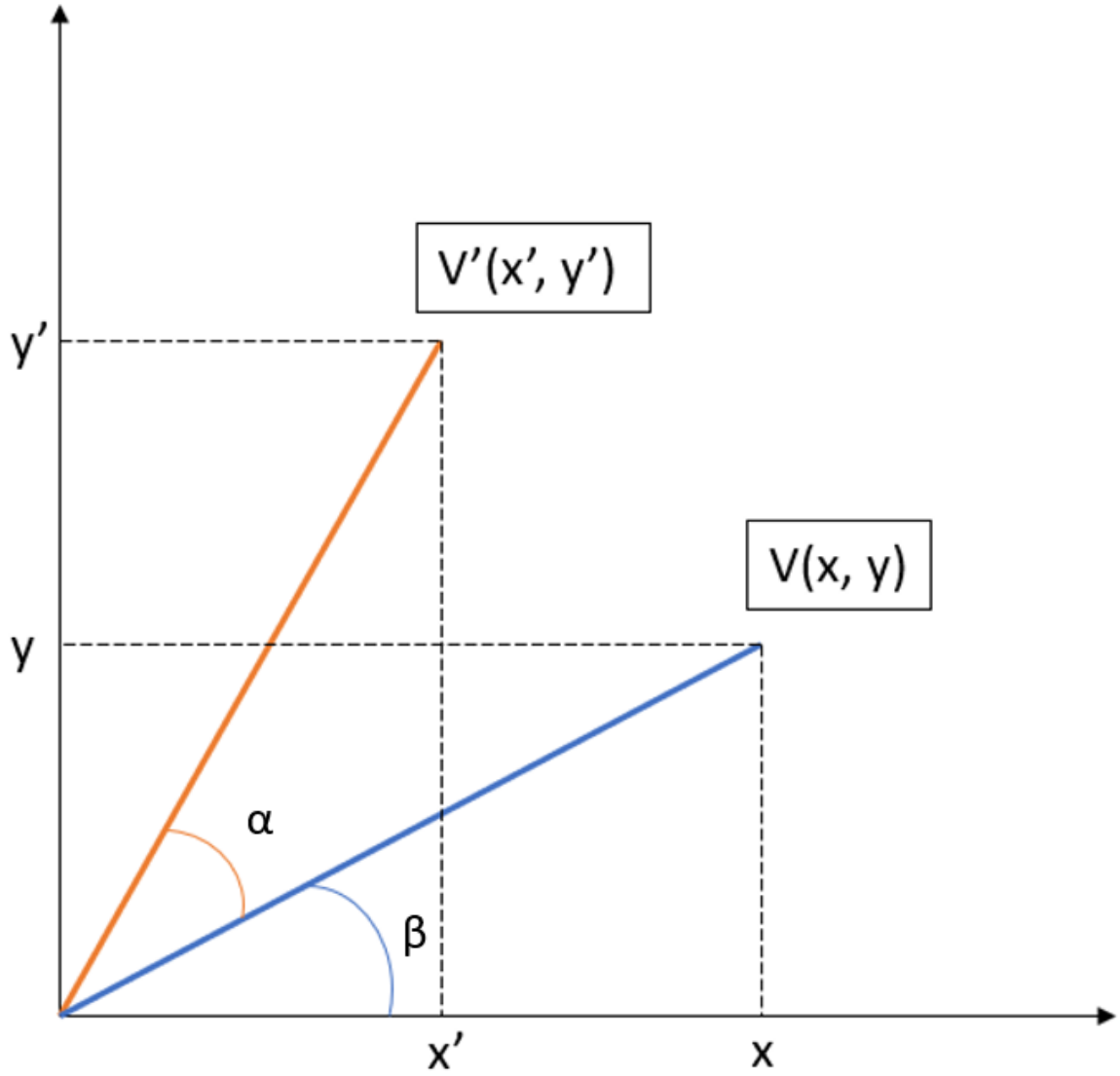


Figure 14: CORDIC Vector rotation

As illustrated in Figure 14, the CORDIC algorithm rotates vector V by an angle α with the coordinates (x, y). Through this process, a new vector V' with coordinates (x', y') is obtained. This is described by equations 2.9 and 2.10

$$x' = xcos(\alpha) + ysin(\alpha) \tag{2.9}$$

$$y' = xsin(\alpha) - ycos(\alpha) \tag{2.10}$$

These equations can be rearranged as depicted in Equations 2.11 and 2.12.

$$x' = cos(\alpha)[x-(ytan(\alpha))] \tag{2.11}$$

$$y' = cos(\alpha)[y + xtan(\alpha)] \tag{2.12}$$

When the rotation angles are limited to $tan(\alpha)$, the tangent term multiplication is simplified to a simple shift operation. Arbitrary rotation angles can be obtained by conducting a sequence of increasingly smaller basic rotations. If the decision at the iteration n is to rotate in which direction rather than whether to rotate, then the expression $cos(\alpha_n)$ becomes a constant, since $cos(\alpha_n) = cos(-\alpha_n)$. Upon this, the iterative rotation can now be written as depicted in equations 2.13 and 2.14, where the rotation direction $d_n = \pm 1$ and the scaling factor $K_n = cos(arctan(2^{-}n))$.

$$x' = K_i[x + yd_n 2^{-n}] \tag{2.13}$$

$$y' = K_i[x - yd_n 2^{-n}] \tag{2.14}$$

The scaling factor can be removed from this equation, as its value is treated as part of the system's processing gain.

The sequence of directions of elementary rotations uniquely defines the angle of a composite rotation. A decision vector can be used to represent this sequence. An angular measuring system based on binary arctangents is the set of all feasible decision vectors. Conversions between this angular system and others are possible with the use of an extra adder-subtractor that collects the elementary rotation angles at each repetition. The angle computation block extends the CORDIC method with a third equation, as seen in equation 2.15.

$$z_{n+1} = z_n - d_n arctan(2^{-n}) \tag{2.15}$$

The angle computation block is initialized with the desired rotation angle while in rotation mode. The rotation choice is taken at each iteration to reduce the magnitude of the residual angle in the angle computation block. Each iteration's choice is therefore dependent on the sign of the residual angle after each step. The rotation mode CORDIC equations are shown in equations 2.13, 2.14 and 2.15, where $d_n = -1$ if $Z_n < 0$, and $+1$ if $Zj > 0$, such that z is iterated to zero. These equations provide the results depicted in equations 2.16, 2.17, 2.18 and 2.19, after n iterations, where G_n represents the gain

and A represents the rotated angle, which can be expressed as depicted in equation 2.20.

$$x' = G_n[xcos(A) + ysin(A)] \tag{2.16}$$

$$y' = G_n[xcos(A) - ysin(A)] \tag{2.17}$$

$$\prod_{i=0}^{n-1} \sqrt{1 + 2^{-2n}} \tag{2.18}$$

$$A = \alpha - z_n \tag{2.19}$$

$$\sum_{i=0}^{n-1} d_n arctan(2^{-n}) \tag{2.20}$$

Because the tangent in the first iteration is 2^0, the CORDIC rotation method is limited to rotation angles between $-\pi/2$ and $\pi/2$. An initializing rotation is necessary for composite rotation angles greater than $\pi/2$. For example, if rotations with rotation angles between $-\pi$ and π are wanted, a beginning rotation of $\pm\pi/2$ is required.

The FFT complex twiddle factor multiplications can be eliminated by transforming them into these CORDIC operations, consequently reducing the twiddle factor memories. The use of CORDIC in FFT results in the elimination of multipliers, saves area, power and cost. CORDIC finds many applications as it provides a simpler way of computing complex multiplications. It is proved that CORDIC is most suitable alternative [25].

Xilinx provides as well a CORDIC IP Core with multiple configurations, which are described below in Chapter 3.

2.6 EMI and EMC

Electromagnetic compatibility (EMC) and electromagnetic interference (EMI) are frequently referred to when discussing the regulatory testing and compliance of electronic and electrical products.

Electromagnetic compatibility and interference are critical design factors, meaning that if these design factors are not addressed in the early stages of product development, this might result in time-consuming and cost requirement to redesign the product. If the developed product fails to pass regulatory certifica-

tions, it is prevented from going to market until those concerns are handled [26].

2.6.1 EMI

EMI is the interference caused by an electromagnetic disturbance that affects the performance of a device. Sources of EMI can be environmental, such as electrical storms and solar radiation, but more usually will be another electronic device or electrical system. If the interference is in the radio frequency spectrum, it is also known as radio frequency interference or RFI.

2.6.2 EMC standards

The device to be developed will be used in the automotive industry. As a result, understanding some of the EMC standards required in this industry is critical.

EMC is a measure of a device's ability to operate as intended in its shared operating environment while, at the same time, not affecting the ability of other equipment within the same environment to operate as intended.

Automotive standards addressing electromagnetic compatibility (EMC) are developed mainly by CISPR, ISO and SAE. CISPR and ISO are organizations that develop and maintain standards for use at the international level. SAE develops and maintains standards mainly for use in North America. In the past, SAE developed many EMC standards which were eventually submitted to CISPR and ISO for consideration as an international standard. As the SAE standards become international standards, the equivalent SAE standard is then withdrawn as a complete standard and reserved for use to document differences from the international standard.

CISPR/D is responsible for developing and maintaining the standards used to measure the emissions produced by vehicles and their components. ISO/TC22/SC32/WG3 is responsible for developing and maintaining the standards used for immunity testing of vehicles and their components. ISO standards for the vehicle industry are mainly broken into two categories, vehicle (ISO 11451-xx) or component (ISO 11452-xx, ISO 7637-xx) [4].

2.6.3 CISPR-25 standard for automotive industry

CISPR-25 standard deals with radio disturbance characteristics for the protection of receivers used on-board vehicles, boats and on devices. This standard is commonly used by the vehicle manufacturers to assure good performance of receivers mounted on-board the vehicle. If the radio mounted in the vehicle,

boat or other device does not perform reliably, then consumer satisfaction and ultimately product sales could suffer.

CISPR-25 has two parts. One part deals with a full vehicle or system test in which the antennas mounted on the vehicle are used to sense the noise generated by the different electric and electronic systems mounted on the same vehicle. This test shows how much noise generated by the vehicle will be introduced into the radios antenna port (sort of a self-immunity test). The other section of the standard deals with conducted and radiated measurements of vehicle components and modules. In this section, the module of radiated emissions test section of CISPR-25 is going to be described, and only briefly highlight some of the additions needed to support electric vehicles. More specifically, this article will concentrate on the chamber requirements for the standard [4].

CISPR-25 states that the electromagnetic noise level in the test area has to be 6 dB lower than the lowest level being measured. Some of the radiated emissions limits found in CISPR-25 are as low as 18 dB (μV/m). This means that the ambient noise must be 12 dB (μV/m) maximum for a compliant environment. An RF shielded room is typically used to keep RF signals from the external environment out of the test area so that the Equipment Under Test (EUT) remains the dominant source of any radiated interference [4].

One of the most efficient and cost effective is a polystyrene based absorber that combines a high-performance ferrite tile with a polystyrene EMC absorber, having 60cm x 60cm base and 60cm height. The main absorber substrate is based on expanded polystyrene (EPS), which is volumetrically loaded with lossy materials, and environmentally friendly fire retardants. Advanced uniform loading in the manufacturing process results in superior RF performance an excellent absorption uniformity. The closed cell structure of this type of absorber makes it suitable for use even in high humidity environments. These features all contribute to providing for a better controlled and predictable chamber test environment. The performance of one type of hybrid polystyrene absorber is depicted in Figure 15.

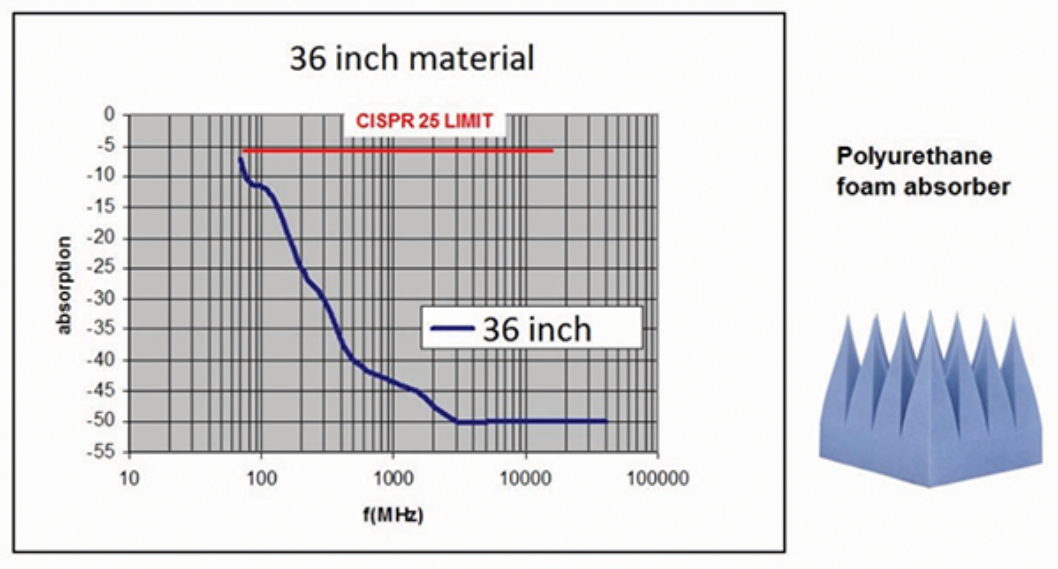


Figure 15: Performance of a 36" polyurethane absorber material [4]

The layout and dimensions of the typical CISPR 25 anechoic chamber is guided by the standard. Several guidelines must be followed when sizing the chamber and the starting point is the EUT, which determines the size of the test bench. Figure 16 depicts the anechoic chamber's setup.

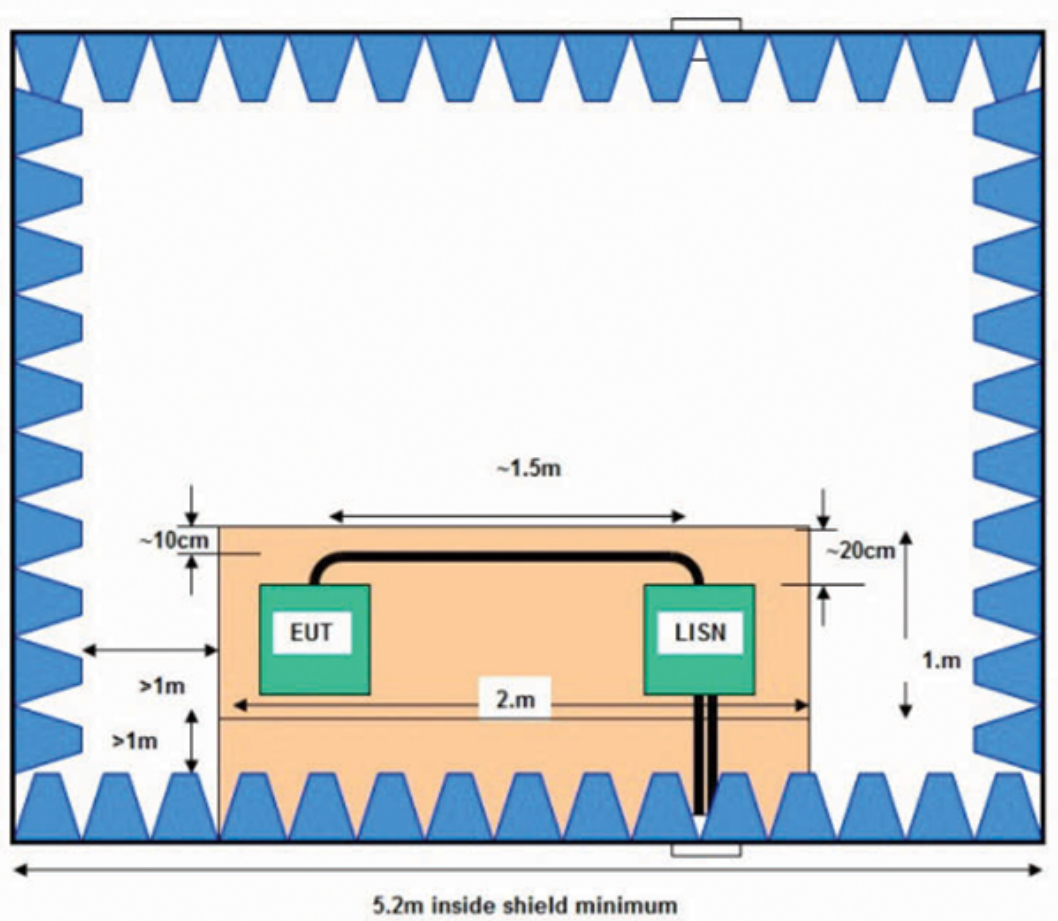


Figure 16: Anechoic Chamber and test bench setup [4]

The bench must accommodate the largest EUT and all the cables that are needed to power and communicate with the device. Regarding the power setup, a Line Impedance Stabilization Network(LISN) is required in order to perform EMC tests, as it isolates power line mains from EUT and it isolates noise produced by the EUT from getting coupled to power line mains, leading to a clean power delivered to the EUT. The cables are routed in a cable harness that is positioned along the front edge of the bench. The cable harness itself is a significant component of the EUT and is the main component illuminated by the measurement antenna since at lower frequencies (frequencies for which the device under test is electrically small) the main coupling to radiated fields will occur through the cables feeding the device [4].

Chapter 3: Analysis and design

3.1 Resources

In this section, all of the hardware and software resources needed and used to successfully complete the project are specified.

3.1.1 Hardware

Zedboard development Kit

ZedBoard is a full development kit for designers who want to experiment with projects that use the Xilinx Zynq-7000 All Programmable SoC. The board has all of the interfaces and supporting functions required to allow a wide range of applications. The board's expandability makes it perfect for quick prototyping and proof-of-concept development [5].

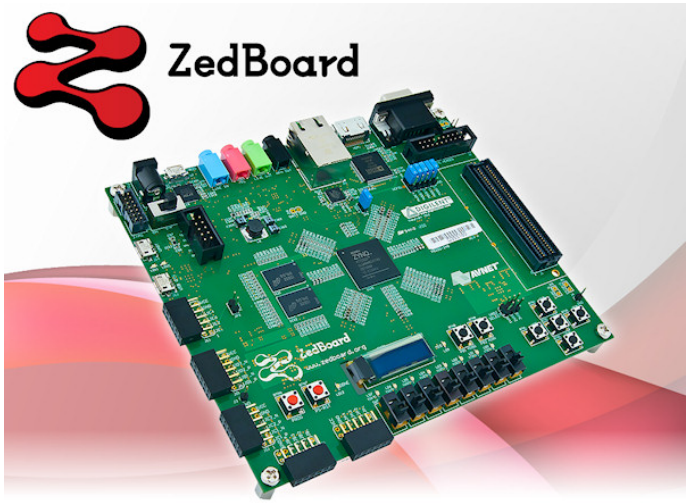


Figure 17: Zedboard Development Kit [5]

AD9467 FMC evaluation board

The AD9467 is a 16-bit, monolithic, Intermediate Frequency (IF) sampling Analog to Digital Converter (ADC) with a conversion rate of up to 250 Megasamples per Second (MSPS). The board contains three inputs: (1) an analog input, (2) a differential amplifier input and (3) a clock input. It also provides options

to drive the clock and analog inputs of the ADC. This can be done by programming the AD9517 clock chip and setting up the ADL5565 differential amplifier respectively [6].

Figure 18: AD9467 FMC Evaluation Board [6]

The AD9467 evaluation board provides all of the support circuitry required to operate the AD9467 in its various modes and configurations.

3.1.2 Software

Vivado IDE

Vivado is a front-end design simulation tool provided by Xilinx, that includes the Vivado Design Suite software tool, used for synthesis and analysis of HDL designs with additional features for system on a chip development and high-level synthesis.

Vitis

Vitis is a software development environment for FPGAs that combines many Xilinx tools, including the Xilinx SDK, Vivado High-Level Synthesis (HLS), and Software-Defined System On Chip (SDSoC). The functionality of each of them has been consolidated under Vitis unified software platform, and it enables the development of embedded software and accelerated applications on heterogeneous Xilinx platforms. This software development environment is used to develop software for the PS.

Visual Studio

Microsoft Visual Studio is a Microsoft Integrated Development Environment (IDE). It is used in the creation of computer programs, websites, web applications, online services, and mobile apps. Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store, and Microsoft Silverlight are used by Visual Studio. It can generate both native and managed code. This IDE was chosen to develop the project's Graphics User Interface (GUI), using C# programming language and Windows Forms.

3.1.3 IP Cores

For this project's hardware component, four main IP Cores that belong to the PL were chosen. This section is composed by a brief description of each one, as well as the communication protocol between the PL and the PS. The configurations made on each IP Core described in this section will be explained further in this document in Chapter 4.

DDS Compiler IP Core

Modern digital communication systems rely on direct digital synthesizers (DDS) or Numeric Controlled Oscillators (NCO). For instance, quadrature synthesizers are used to build digital down and up converters, demodulators, and modulation schemes of various types. A lookup table scheme is a common method for digitally generating a complex or real valued sinusoid. The lookup table stores sinusoid samples, and a digital integrator is used to generate a suitable phase argument, which is then mapped to the desired output waveform by the lookup table [27].

Fast Fourier Transform IP Core

The Fast Fourier Transform IP Core computes an N-point Discrete Fourier Transform (DFT) or Inverse Discrete Fourier Transform (IDFT) [9]. This IP core receives a bit vector with real and imaginary components as input. The output vector is represented using bits for each of the real and imaginary components of the output data. Input data is presented in natural order and the output data can be in either natural or bit reversed order. The complex nature of data input and output is intrinsic to the FFT algorithm, not the implementation. The FFT IP core accepts complex data samples, but it can perform a transform on real-valued data by setting all imaginary input samples to zero. In this IP Core there are three arithmetic options available to compute the Fast Fourier Transform: (1) Full-precision unscaled arithmetic, (2) Scaled

fixed-point, where the scaling schedule is provided, and (3) Block floating-point (run time adjusted scaling).

CORDIC IP Core

The CORDIC IP core implements a generalized coordinate rotational algorithm that solves iteratively trigonometric and hyperbolic equations, square roots and converts between rectangular and polar coordinates [7]. Since the CORDIC algorithm introduces a scale factor to the amplitude of the result, this IP Core provides the option of automatically compensate the CORDIC scale factor.

The CORDIC IP Core was used to calculate the magnitude of the signal output of the FFT, given it's complex value. There are two ways to calculate magnitude using CORDIC IP Core. One of them is calculating the absolute value by using multipliers and adders on the FFT's real and imaginary component, and then use CORDIC to calculate the square root of the absolute value. The other is using vector translation. As shown in figure 19 when the vector translational functional configuration is selected, the input vector (X,Y) is rotated using the CORDIC algorithm until the Y component is zero. This generates the scaled output magnitude and the output phase.

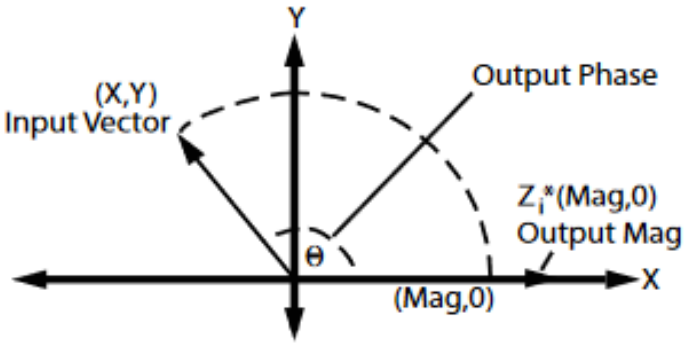


Figure 19: Vector translation [7]

Vector translation was applied to calculate magnitude values in this simulation. It is essential to correctly configure the CORDIC IP Core in order to obtain the correct vector translation and hence obtain the correct magnitude values for the signal. The exact configuration of the CORDIC IP Core is explained further in this document on Chapter 4.

FIR Compiler IP Core

The Finite Impulse Response (FIR) Filter is one of the most ubiquitous and fundamental building blocks in DSP systems. Although its algorithm is simple, the implementation specifics can be vast and time consuming for hardware developers today, especially in filter-dominated systems such as Digital

Radios. The FIR Compiler IP Core provides a common interface to generate highly parameterizable, area-efficient high-performance FIR filters using either Multiply-Accumulate (MAC) or Distributed Arithmetic (DA) architectures. This allows to manage trade-offs between different hardware architectures of their FIR Filter specification, reducing filter implementation time [28]. This IP Core is important in this project and its purpose will be discussed further.

3.2 Requirements and constraints

The goals of the project dictate the functional requirements, which are: (1) capture signals and calculate its spectrum, (2) hardware accelerated spectrum calculation, (3)provide high resolution, (4)the ability to capture signals between 9kHz and 108MHz and (5)display the captured signals through a user interface.

As for qualities of the system, which compose the non-functional requirements, is that this is a deterministic system.

Regarding the development constraints of the project, they are divided in technical and non-technical. The technical are: (1) use Zedboard development kit, (2) use EDA tools as Vivado, (3) Vitis for the development of the software component, (4) Verilog, (5) C programming language, and(6) C# for the user interface application implementation.

As for non-technical, the project was developed in a company with a 10 month period to finish the project.

3.3 ADC clock source

The AD9467 Evaluation Board provides three possible clock sources for clocking the AD9467:

- External Clock Source (default);
- Crystal Oscillator;
- Clock generator with integrated VCO (AD9517).

To change the clock source, the evaluation kit board requires some hardware modifications, that is, soldering and de-soldering of some components on the AD9467 Evaluation Board. The external clock source requires a function generator, which may not be convenient. The crystal oscillator cannot be used either, because it can only reach a frequency of 25 MHz, and according to Nyquist's Theorem, this would

mean that it would only be possible to see the signal's frequency component between 1Hz and 12.5MHz. Lastly, the AD9517 is a clock generator with an integrated 1.6 GHz VCO. With this clock generator, it is possible to reach a clock frequency of 250 MHz.

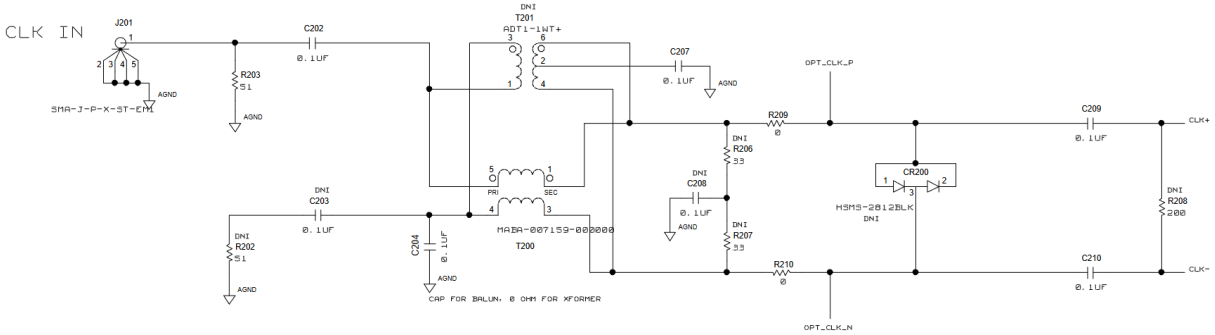


Figure 20: External Clock Source (Default) [6]

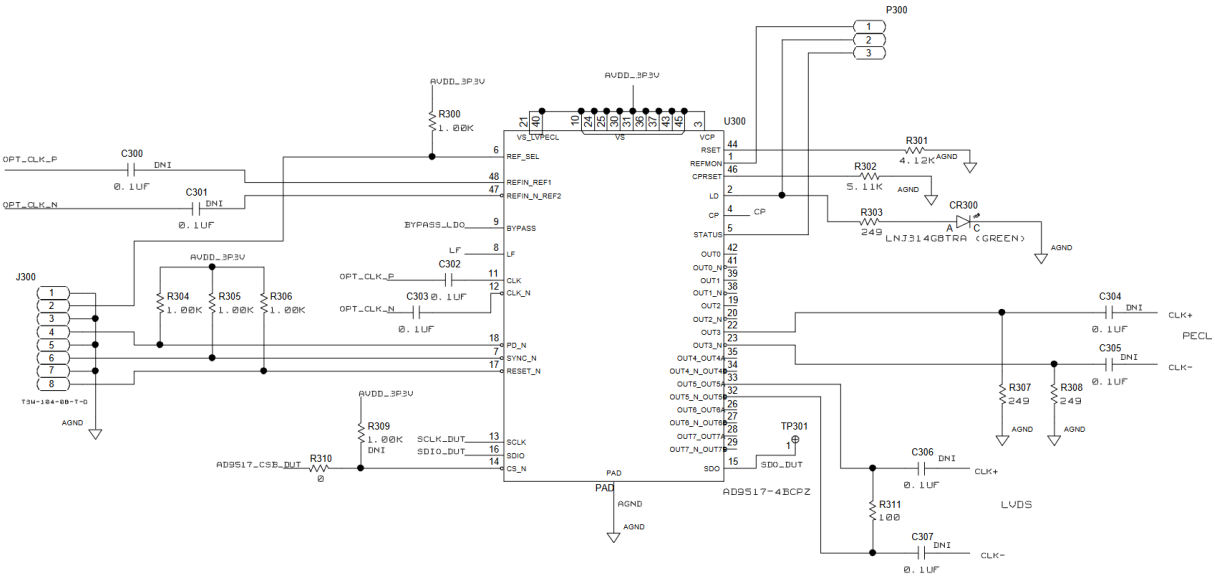


Figure 21: AD9517 Clock Source [6]

Following tests with the AD9517 clock source, it was determined that its use was not ideal due to variations in its output frequency with each test. Every test produced a different output value that fluctuated by less than 3MHz from the intended frequency. Following this conclusion, the external clock was used.

3.4 System architecture

Regarding the system's limitations, despite the FFT IP Core's ability to compute a maximum of 65535 points, Zedboard can only compute 16384 points due to a lack of BRAM. Therefore, it is impossible to meet the spectrum analyzer's resolution bandwidth requirements with such number of FFT points. Therefore, it

is impossible to meet the spectrum analyzer's resolution bandwidth requirements with this number of FFT points because, according to equation 3.1, which calculates the resolution bandwidth (RBw) by dividing the sample clock frequency (F_{CLK}) by the number of FFT points(N_{FFT}), with a clock frequency of 250 MHz, the resolution bandwidth value is 15.258 kHz.

$$RBw = \frac{\frac{F_{CLK}}{2}}{\frac{N_{FFT}}{2}} = \frac{F_{CLK}}{N_{FFT}} \tag{3.1}$$

In order to get a higher resolution, by lowering the RBw value, the solution is either changing the clock frequency or increasing the number of FFT points. As it is impossible to increase the number of points due to the hardware's limitations, the solution is to decrease the clock frequency. But then, given the Nyquist theorem, this would reduce the spectrum span, and since there's also a requirement to have a span of 108 MHz, a strategy is needed to overcome this problem, by implementing an heterodyne architecture.

3.4.1 Heterodyne architecture

In a heterodyne architecture, as depicted in Figure 22, the input signal is mixed with a tone carrier, denoted as Local Oscillator.

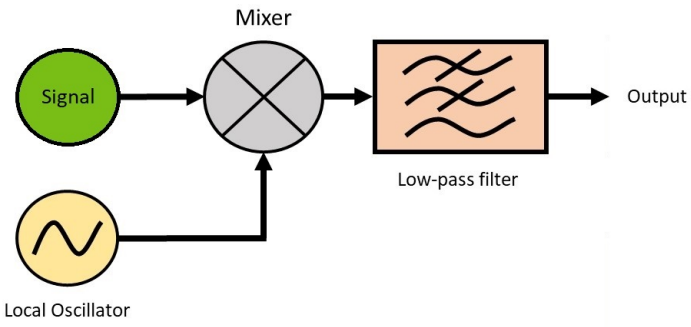


Figure 22: Heterodyne architecture

The mixer's output signal is the input spectrum shifted right by the oscillator's frequency value. This means that the first value in the spectrum's frequency axis corresponds to the local oscillator's frequency. Figure 23 illustrates the result of this architecture.

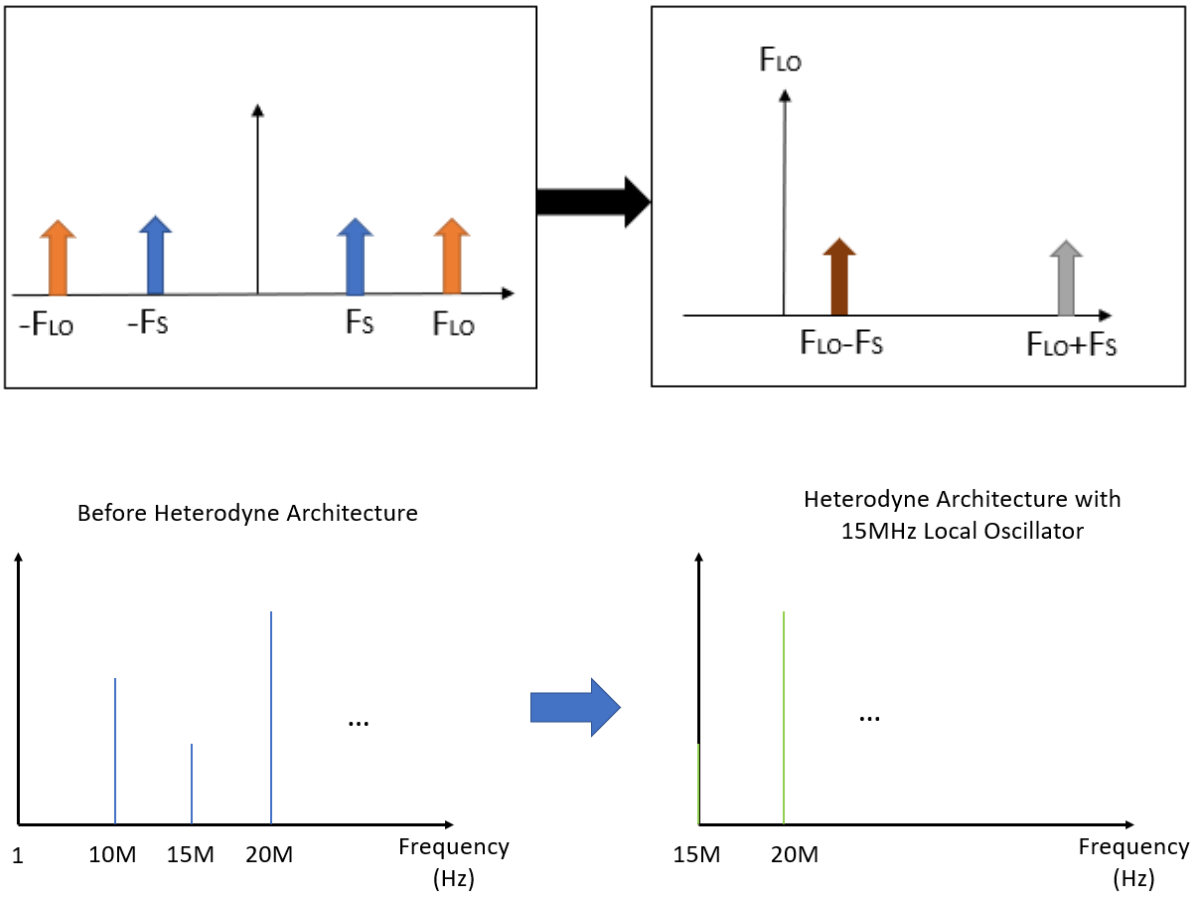


Figure 23: Heterodyne Arquitecture Example

A filter must also be implemented to properly use this architecture in order to avoid aliasing, and because when two signals are mixed, two components of those signals are produced, one at the sum of the two signals and another at the difference of the two signals. The sum of the two signals is filtered using a Low-Pass Filter because only the difference between the two signals is required for this application. This signal processing technique is usually used to shift between frequency ranges.

3.5 Use case

Regarding the use case, as illustrated in Figure 24, the user selects the range of frequencies to be analyzed. The spectrum analyzer acquires the data related to conducted emissions, processes the data by performing the calculations necessary to obtain the spectrum, and finally it send the processed data to the computer, for the user to check and analyze it.

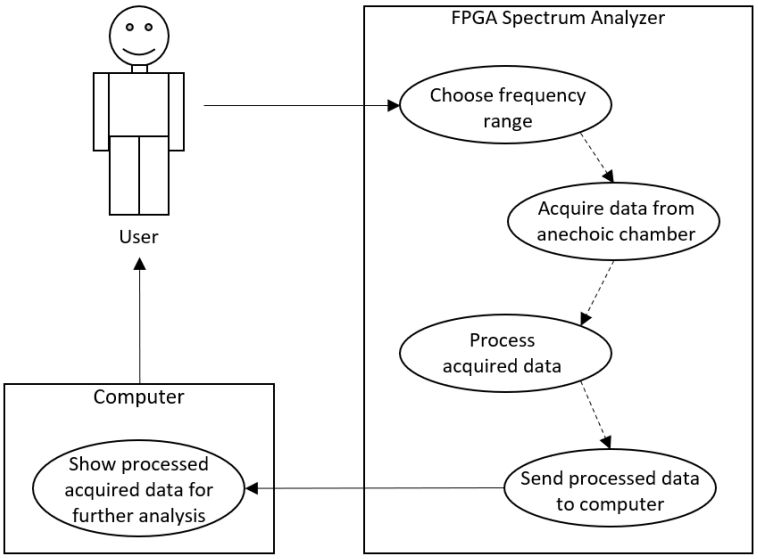


Figure 24: Use Case

The spectrum analyzer is connected to an anechoic chamber, which is designed to stop reflections of electromagnetic waves and it is isolated from energy entering from their surroundings. The setup used in the anechoic chamber, has great influence in the measured results. CISPR 25 [4] describes and illustrates the correct setups where conducted and radiated emissions are measured. If the setup used doesn't comply with the EMC standard, the test results will be wrong for sure. After obtaining the results, these need to be within the EMC norm. If not, the root cause must be investigated.

Chapter 4: Implementation

4.1 Data acquisition

The first step in any endeavor like the one tried in this thesis development, is always to understand the several components that make up the system and understand their characteristics and how they operate, so that they may be made to interact with one another, resulting in a more complex device. The first and most relevant element of this system is the front-end acquisition interface. For this thesis, as described earlier, a high-speed 16-bit ADC was chosen, the AD9467 from Analog devices, which is a 16-bit 250 Msps ADC. Analog Devices provides a reference design for the AD9467 ADC frontend evaluation board to run on the Zedboard. This reference design is composed by the hardware and software components. The block diagram from Vivado is illustrated in figure 25.

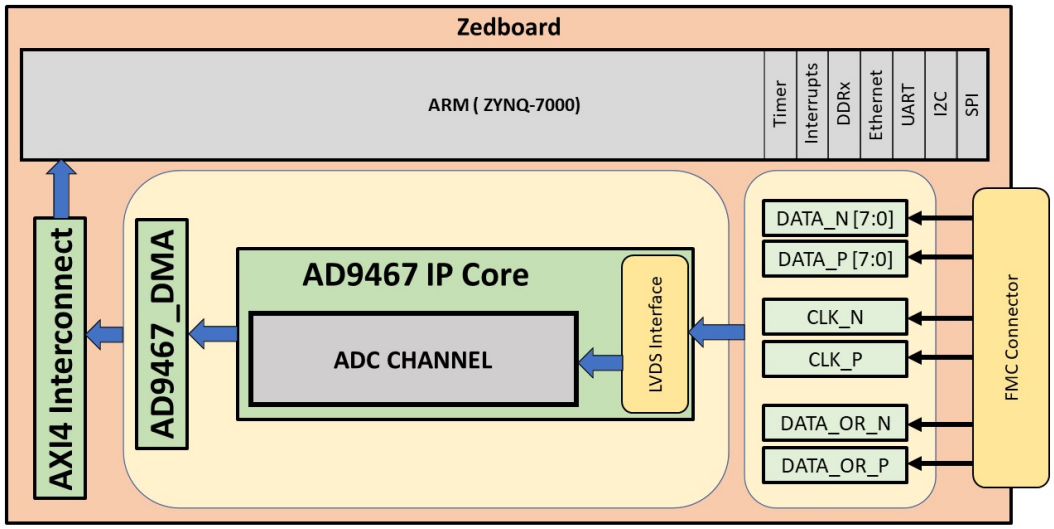


Figure 25: AD9467 Reference Design Diagram Illustration

The main hardware components responsible for the data acquisition are the AD9467 IP Core, composed by the ADC inputs, and the DMA IP core from Analog Devices, depicted in Figure 26. Aside from that, the AXI4 communication protocol used between the Processing System (PS) and the Processing Logic (PL) is a key component on the block diagram.

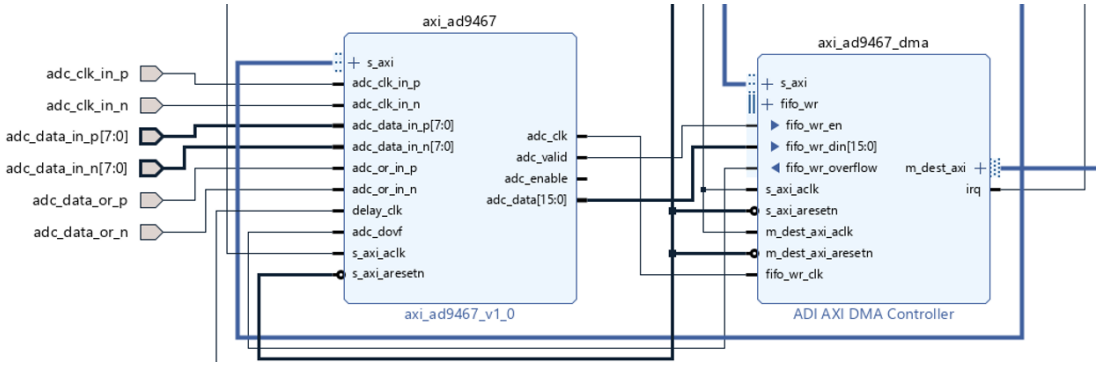


Figure 26: AD9467 Signal Acquisition IP Blocks

4.1.1 Sampling

The AD9467 IP Core is used to interface the AD9467 ADC. It has the inputs for the signal, the external clock and the differential amplifier. These inputs are mapped to the FMC connector pins in a constraints file on Vivado. Regarding the outputs, it has: (1) a data output that has an intergrated FIFO Interface, (2) a valid signal that indicates that there is valid data available on the bus, and (3) the clock output, which is the external clock fed to the AD9467. Table 1 depicts in detail the interface, inputs, outputs and a brief description of each one.

Interface	**Pin**	**Type**	**Description**
ADC interface	**ADC Interface Signals**		
	adc_clk_in	input	LVDS input clock
	adc_data_in*	input[7:0]	LVDS input data
	adc_or_in*	input	LVDS input over range
Delay Interface	**Interface used to control the delay lines**		
	delay_clock	input	Clock used by the IDELAYCTRL.
S_AXI	**AXI Slave Memory Map interface**		
ADC FIFO	**FIFO interface for connecting to the DMA**		
	adc_clk	output	The input clock is passed through an IBUFGDS and a BUFG primitive and adc_clk reults. This is the clock domain that most of the modules of the core run on
	adc_valid	output	Set when valid data is available on the bus.
	adc_enable	output	Set when the channel is enabled, activated by software.
	adc_data	output[15:0]	Data bus.
	adc_dovf	input	Data overflow input, from the DMA.
	adc_dunf	input	Data underflow input.

Table 1: AD9467 Interface

The AD9467 IP Core, axi_ad9467, instantiates a LVDS interface module, two-channel processing module, ADC common register map, AXI handling interface and a delay control module. The LVDS interface module receives LVDS signals for clock, data[7:0], and over range as inputs and outputs single ended signals. Data signals are passed through an IDELAYE2[1], allowing each sample to be delayed independently via the delay controller register map. This is mainly used to adjust the I/O timing delay, such as adjusting the timing relationship between the ADC acquisition clock and the ADC acquisition data I/O. This is configured later on the software component.

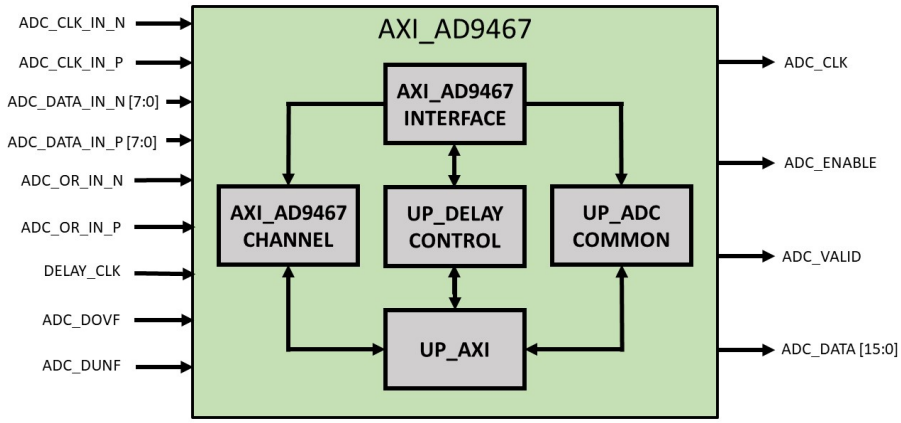


Figure 27: AD9467 IP Core Detailed Architecture

A more detailed structure of the AD9467 IP Core is depicted in Figure 27. The up_adc_common module implements the ADC COMMON register map, which allows basic ADC monitoring and control. A PRBS monitor, data format conversion, a DC filter, and the ADC channel register map are all implemented in the channel module. The raw data received from the interface is analyzed by the PRBS monitor. The delay controller module (up_delay_cntrl) enables dynamic reconfiguration of the IDELAYE2 block, allowing software to execute a calibration operation.

4.1.2 Data management

The outputs from the AD9467 IP Core connect to the DMA IP Core. In the DMA IP Core, the AD9467 IP Core's valid signal is connected to the enable signal of the DMA IP Core, indicating that if there is valid output data, it enables the DMA Core. Then it has the data input connected to the AD9467 IP Core data output. The DMA IP Core sends the values from the ADC to the PS through a transfer order made by the software component. As for the clock, everytime the clock is high, a value read from the ADC is written

[1] IDELAYE2 is a 32-tap, wrap-around programmable delay primitive in every IO block of Xilinx 7 series FPGA

into the DMA. Figure 28 depicts the Analog Devices's AXI_DMA IP Core's block diagram, along with the interface's pins used in Table 2.

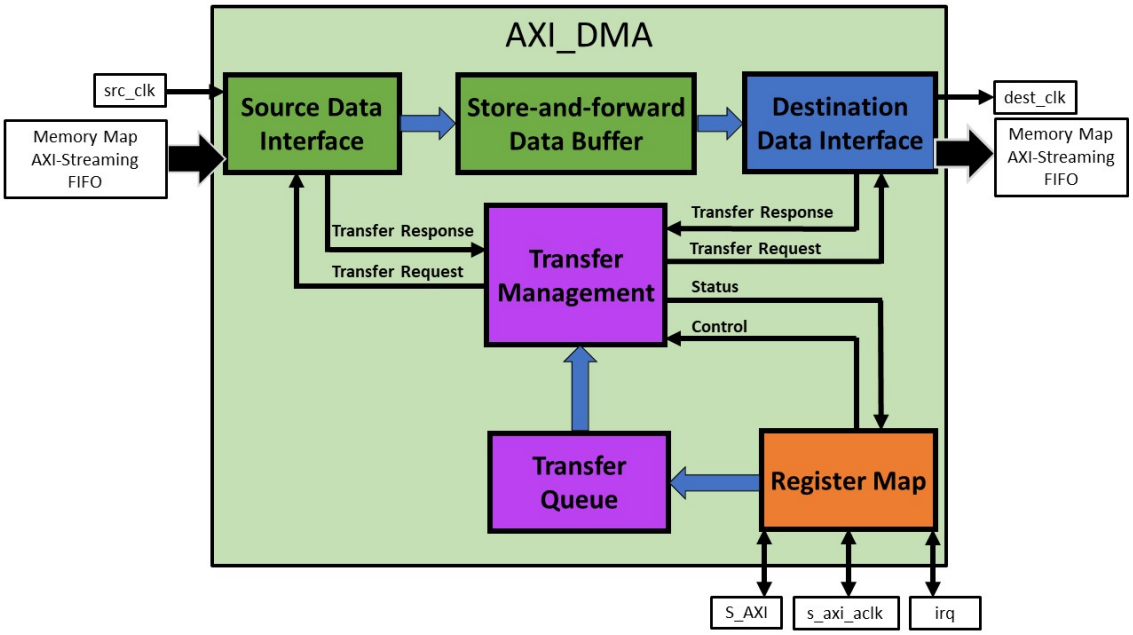


Figure 28: Analog Devices AXI_DMA IP Core Block Diagram

Name	Type	Description
s_axi_clk	Clock	All s_axi signals and irq are synchronous to this clock.
s_axi_aresetn	Synchronous active low reset	Resets the internal state of the peripheral.
s_axi	AXI4-Lite bus slave	Memory mapped AXI-lite bus that provides access to modules register map.
irq	Level-High Interrupt	Interrupt output of the module. Is asserted when at least one of the modules interrupt is pending and enabled.
m_src_axi_aclk	Clock	The m_src_axi interface is synchronous to this clock.
m_src_axi_aresetn	Synchronous active low reset	Reset for the m_src_axi interface.
m_src_axi	AXI3/AXI4 bus master	
m_dest_axi_aclk	Clock	The m_src_axi interface is synchronous to this clock.
m_dest_axi_aresetn	Synchronous active low reset	Reset for the m_dest_axi interface.
m_dest_axi	AXI3/AXI4 bus master	
fifo_wr_clk	Clock	The fifo_wr interface is synchronous to this clock.
fifo_wr	FIFO write interface	
fifo_wr_en	Input	The fifo_wr_en is et when valid data is available on the bus

Table 2: AXI_DMA Interface from Analog Devices

Inside the DMA IP Core, a few configurations are made for the input and output data. For the input data, the following configurations were made:

- Type: FIFO Interface;
- AXI Protocol: AXI4;
- Bus Size: 16 bits.

As for the output data:

- Type: Memory-mapped AXI;
- AXI Protocol: AXI4;
- Bus Size: 16 bits.

An internal buffer is used to store data from the source interface before it is forwarded to the destination once that can accept it. The purpose of the buffer is to even out the rate mismatches between the source and destination. As depicted in the DMA IP Core configuration, the destination is a MM interface and the source a FIFO interface with a fixed data rate. In this case, the intent is to keep the buffer as empty as possible so in case the MM interface is not ready, data can be still accepted from the source without risking an overflow. Complementary, in case the destination was a MM interface and the source was a FIFO interface with a fixed data rate, the buffer would be kept as full as possible so in case the MM interface is not ready, data can be still provided to the destination without risking an underflow.

Lastly, the AXI Interconnect block is used in this section of the block diagram to connect all of these IP Cores to the Processing System, ZYNQ-7000, as illustrated in Figure 29.

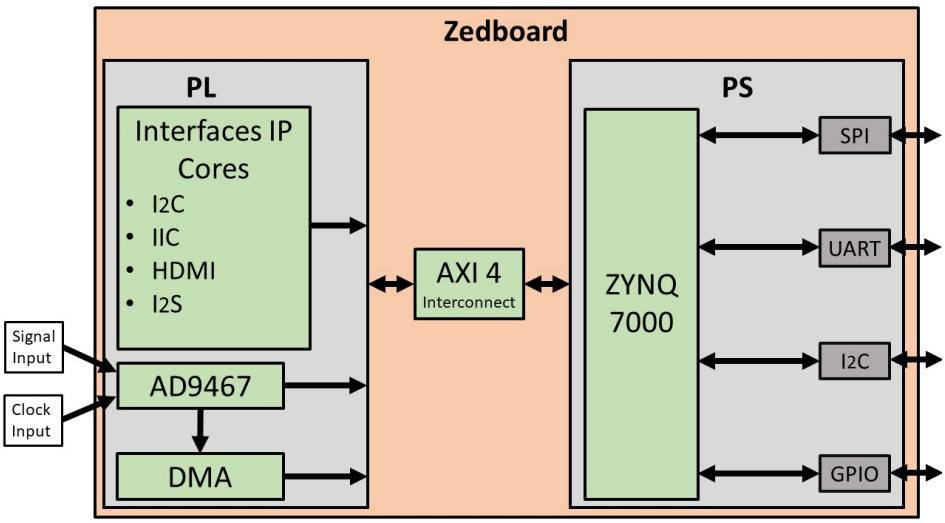


Figure 29: AD9467 Reference Design signal acquisition IP Blocks

4.1.3 AXI4

AXI, or Advanced eXtensible Interface, is one of ARM's AMBA standards. The AXI is a point-to-point interface designed for high-performance, high-speed microcontroller systems.The AXI protocol is built on a point-to-point connection, which avoids bus sharing and allows for more bandwidth and reduced latency. AXI is the most widely used AMBA interface interconnect.

The AXI protocol's essence is that it offers a framework for how distinct blocks inside each chip communicate with one another. It provides a mechanism before any data is sent, ensuring that communication is clear and uninterrupted. As a result, separate components may communicate with one another without any collisions. The AXI protocol is implemented as follows:

- Master & slave must "handshake" to confirm valid signals

- Transmission of control signal must be in separate phases
- Separate channels for transmission of signals
- Continuous transfer may be accomplished through burst-type communication

In this project, AXI4-Lite protocol is used. This protocol's channels connections are depicted in Figure 30 [8].

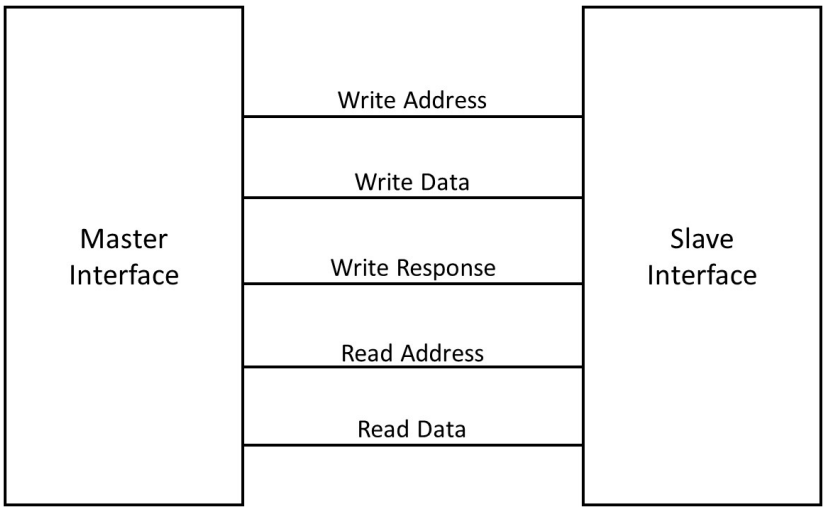


Figure 30: AXI4-Lite Channel connections between master and slave interfaces[8]

The interface works by establishing communication between master and slave devices. There are five separate channels (or more if using an AXI Interconnect Core IP) between these two devices: Read Address, Write Address, Read Data, Write Data, and Write Response. Each channel has its own unique signals as well as similar signals existing among all five. The valid and ready signals exist for each channel as they allow for the handshake process to occur for each channel. In figure 31 it is represented the time diagram of these signals.

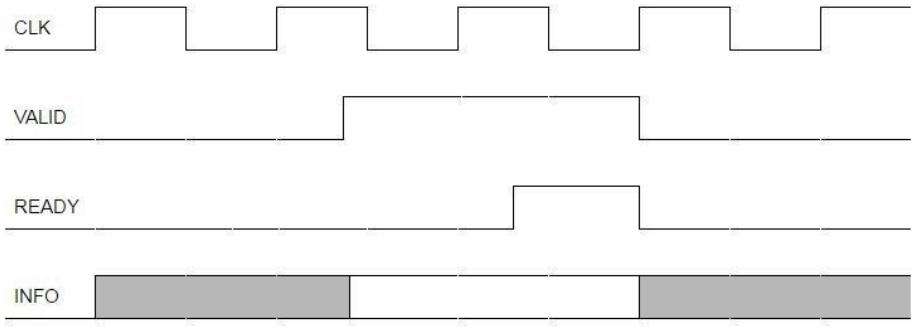


Figure 31: AXI4-Lite READY and VALID signals[8]

As illustrated, information is only transmitted on the first rising edge after Ready and Valid signals are both active. Valid signal indicates when the information, depicted in white, is ready to be sent [8].

Regarding the Processing System, one of the important configurations is the I/O Peripherals, where the UART and SPI are configured. These protocols are fundamental in the system since the ADC communicates using SPI. As for UART, it is used to send the captured data to the computer. Figure 32 shows the Processing System Block Design from Vivado, where an overview of the configurations can be seen, such as the I/O Peripherals, internal clocks, DMA and AXI Protocols.

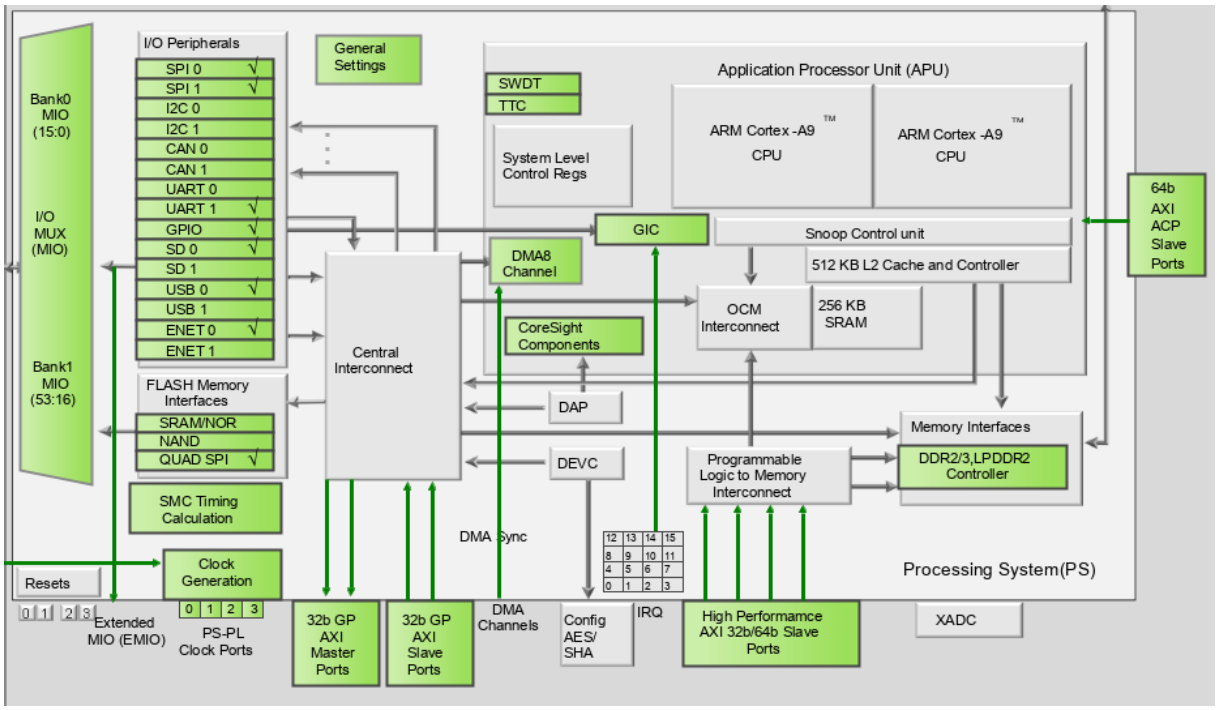


Figure 32: ZYNQ-7000 block design overview

After checking all the configurations, the Chipscope Integrated Logic Analyzer (ILA) IP core was added to the project. This IP Core is a customizable core that can be used to monitor internal FPGA signals in real time and in this manner, the acquired signal can then be analyzed.

After compiling the project, two function generators were connected to the AD9467 Evaluation Board's signal and clock inputs. One was used to provide a clock signal to the board, while the other was used to provide a sinusoidal signal in order to validate the data collected by the ADC. The collected signal could be observed on Chipscope as illustrated in Figure 33.

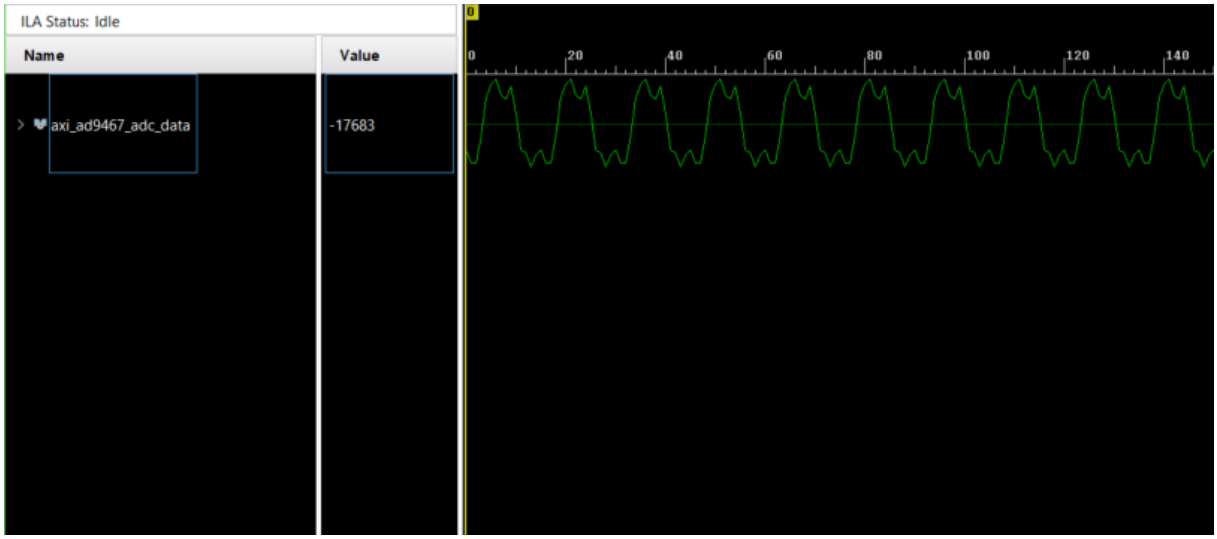


Figure 33: Captured signal on Vivado's Chipscope

As can be observed, the signal is not a perfect sinusoid. This happens because the ADC front-end needs a certain pre-configuration in order to correctly perform the capture. As a result, in order to capture the signal accurately, software must be used to configure the ADC through SPI. As a consequence, following a review of the raw results on Chipscope, the evaluation software code provided by analog devices was compiled and tested on Vitis.

The software component starts by configuring the ADC through SPI. After the configuration, the ADC is tested using a test mode, where it checks data patterns by writing values to the ADC registers and checking if the data matches the written values. The sampling starts and the values read from the ADC are transferred through the DMA to the processor's memory, and they can be seen in the memory map tab from Vitis, shown in figure 34.

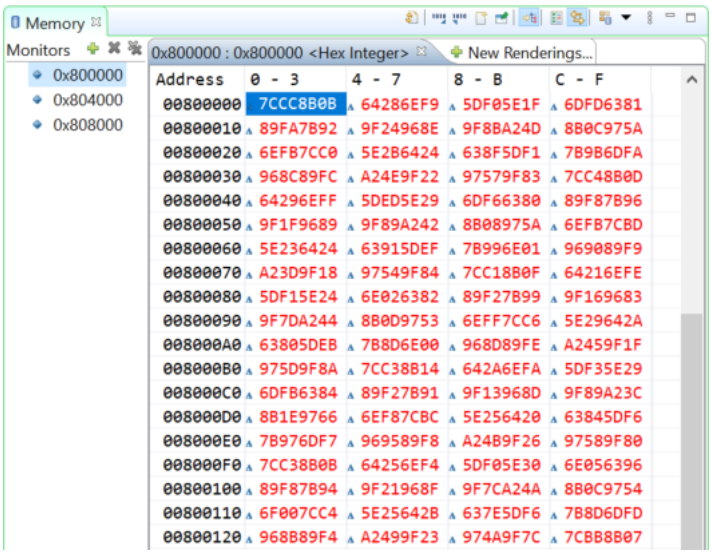


Figure 34: Vitis Memory Map

In order to be able to plot the acquired values from the ADC, code was added to the software to transmit the data shown on the memory map to the computer via UART. Xilinx provides a library, xil_io.h, which includes the interface for the general I/O component, which encapsulates the I/O functions for the processors that do not require any special I/O handling.

Because the ADC has a resolution of 16 bits, the function Xil_In16, which takes a memory address as an argument, was used to read data from the memory locations containing the collected signal samples. To obtain the real voltage values, the raw sample values have to be scaled. To do this, the board's reference voltage, 2.5V, was divided by the ADC's resolution. The real voltage value is the product between this value and the raw value obtained from the ADC. This conversion is presented in Equation 4.1.

$$V_{Signal} = ADC_{RawValue} * \frac{V_{ref}}{2^{Nbit}} \tag{4.1}$$

An infinite cycle was implemented to obtain continuous sampling from the signal. When the last sample is printed, a fresh DMA transfer is initiated and new sample values are written in the same memory region.

Memory addresses are defined on a library created by the hardware project that was compiled on Vivado. The library, xparameters_ps.h, includes the address mappings for the hard peripherals connected to the ARM Cortex A9 core. These address mappings are defined when the PS is configured in Vivado.

After adding the code, a serial port plotter [29] was used to receive and plot the values sent from the board. The test result is shown in figure 35.

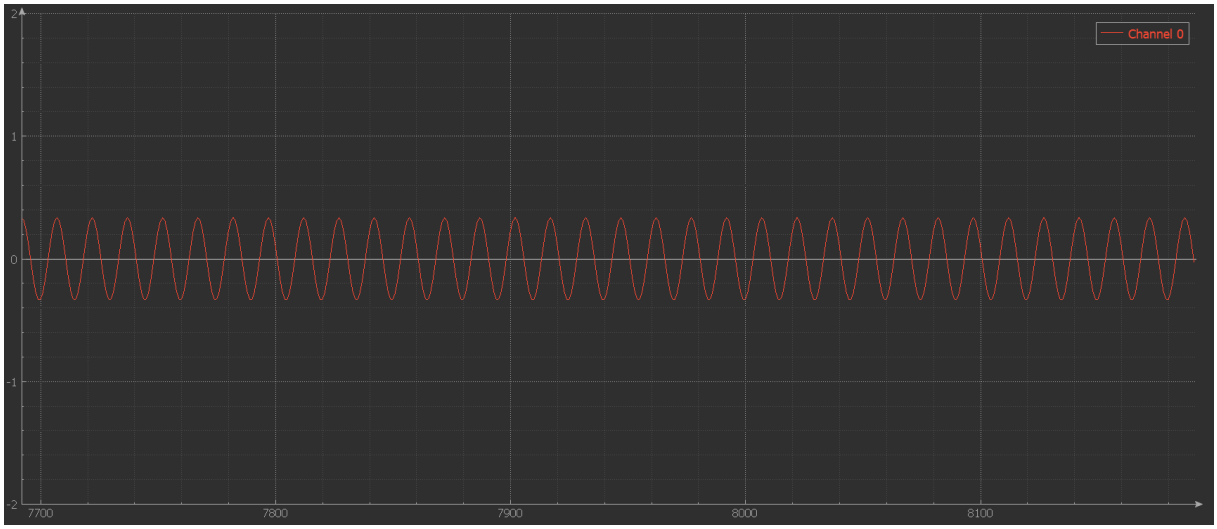


Figure 35: Captured Signal

Lastly, the captured signal with the software running was checked on Chipscope. The results on Figure 36 show that the signal is perfectly captured after the software configures and calibrates the ADC.

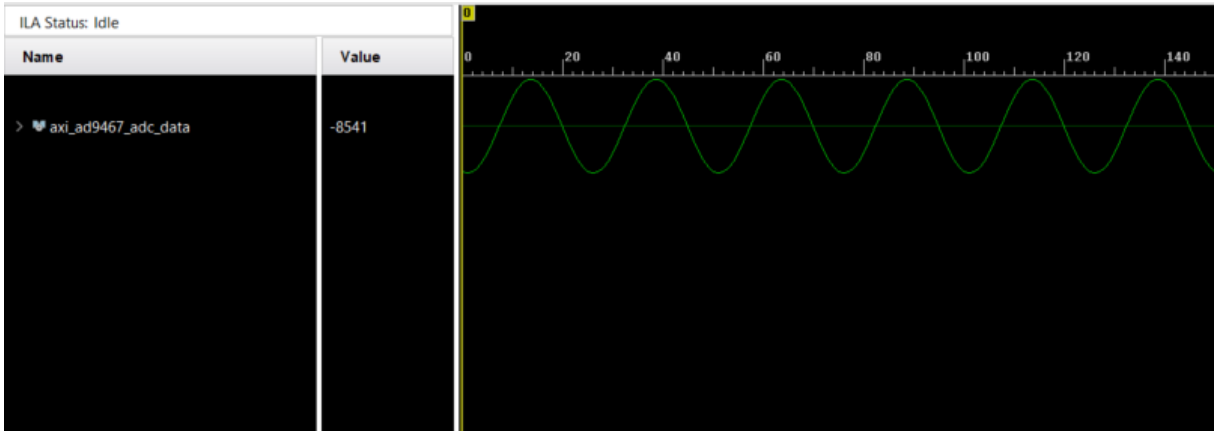


Figure 36: Captured Signal on Vivado's Chipscope after running software

4.2 Data processing simulations

Before beginning to implement the Spectrum Analyzer, it is critical to simulate all of the components that provide the means to build the spectrum from the ADC signal. Five IP Cores were used in these simulations: (1) DDS Compiler, (2) FIR Compiler (3) Fast Fourier Transform, (4) CORDIC, all supplied by LogiCore, and System Clock Generator supplied by Xilinx.

4.2.1 AXI4-Stream

Regarding the IP Cores mentioned, they function exclusively with AXI4-Stream. This was no problem since AXI4-Stream and AXI4 both have the same data, ready and valid input and output signals, as depicted in Figure 37.

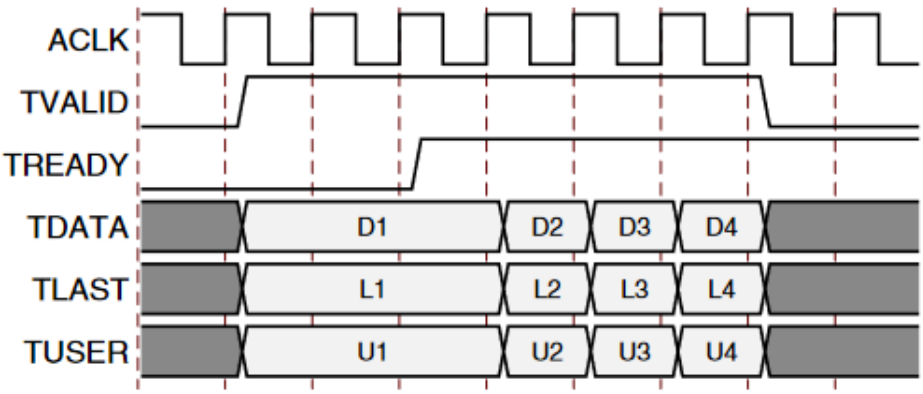


Figure 37: AXI4-Stream READY and VALID signals [9]

TVALID is controlled by the source (master) side of the channel in AXI4-Stream, whereas TREADY is controlled by the receiver (slave). The value in the payload fields (TDATA, TUSER, and TLAST) is valid when TVALID is set. The slave is ready to accept data when TREADY is set. A transfer happens when both TVALID and TREADY are TRUE in the same cycle. The master and slave, respectively, set TVALID and TREADY for the next transfer [9].

4.2.2 Spectrum calculation

Using the IP Cores described in the system resources, a small acquisition system was implemented in order to simulate the system used to calculate the spectrum and magnitude of the signal. The block diagram of that system is shown in 38.

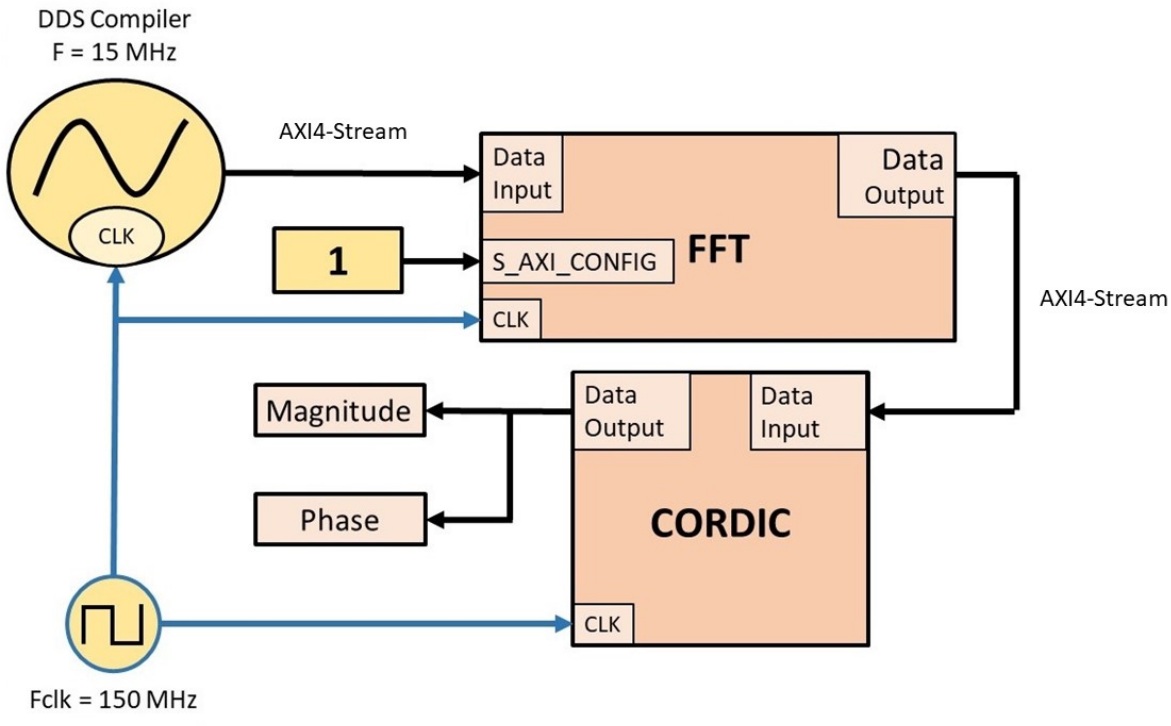


Figure 38: Block diagram of FFT and magnitude calculation simulation circuit

For the clock signal, Xilinx provides an IP Core, Simulation Clock Generator, that generates a clock for simulation purposes only. This clock was set to a frequency of 150 MHz. The source signal was emulated with DDS compiler.

All the IP Cores can be setup by using its configuration window, although some configurations may need to be applied to the IP Cores by writing to the configuration input's array. This will be addressed further below in the FFT IP Core configuration.

DDS Compiler configuration

On the DDS Compiler, the following configurations were made:

- System Clock: 150 MHz;
- Number of Channels: 1;
- Output Width: 15 bits;
- Output Frequency: 5 MHz;

Fast Fourier Transform configuration

On the Fast Fourier Transform IP Core, the following configurations were made:

- Target Clock Frequency: 150 MHz;
- Number of Channels: 1;
- Transform Length: 16384;
- Architecture: Radix-4;

It is important to additionaly setup this IP Core using the S_AXIS_CONFIG input in order to use it successfully. This input has several important settings. The IP core may be customized by writing on this array, and the array width varies dynamically depending on the configurations made in the configuration window. For example, if the scaling choices on the configuration window are set to Scaled, the array size will grow by adding bits that correspond to the scaling factor that must be selected based on the required scaling factor. Depending on the settings in S_AXIS_CONFIG, the FFT IP Core computes Forward Fast Fourier Transforms and Inverse Fast Fourier Transforms. To set the forward FFT, the LSB of the S_AXIS_CONFIG array must be set to '1', hence a constant with the value '1' is placed on this input.

Lastly, some final configurations were made in the FFT IP Core:

- Data Format: Fixed-Point;
- Scaling Options: Unscaled;
- Input Data Width: 32 bits (16 bits for the real component and 16 bits for the imaginary component;)
- Output Order: Natural Order;
- Optional Output Field: XK_INDEX

The maximum transform length available is 65535, but because of the Zynq-7000 resources, in this system, it is only possible to choose 16384, which is equivalent to $\frac{N_{FFT}}{2}$. This has a direct impact on the capability to achieve the resolution bandwidth requirements defined for this project, which can be proven by using Equation 3.1.

Regarding the scaling options, there are three possibilities. The Block Floating-Point configuration dynamically scales the output, which means that at each stage of the algorithm, it determines how much scaling is required to make the most use of the available dynamic range and provides the scaling factor as

a block exponent. This block exponent is added to the FFT IP Core as an output to display the amount of the scaling factor in a calculated input value. In the Scaled configuration, a scaling factor must be specified and given to the FFT IP Core so that it may be applied during the FFT computation. This configuration is not ideal for this project since its objective is to evaluate random signals, and determining the optimal scaling factor would take trial and error, and it would not be optimized because the scaling factor may change based on the input value. The Unscaled configuration was chosen since the output values do not require scaling, which means that all integer bit growth is transported to the output. As a result, additional FPGA's resources are consumed, since this configuration provides full precision.

CORDIC configurations

Regarding the CORDIC IP Core, the following configurations were made:

- Functional Selection: Translate;
- Input and Output: 16 bits each;

Spectrum calculation results

After configuring all the system blocks, a behavioral simulation was ran and the results are depicted in Figure 39. In this simulation, four signals were added. The first signal is from the DDS Compiler, the second signal is the CORDIC output and the third and fourth signals are the control signals from the FFT IP Core *tvalid* and *tuser*.

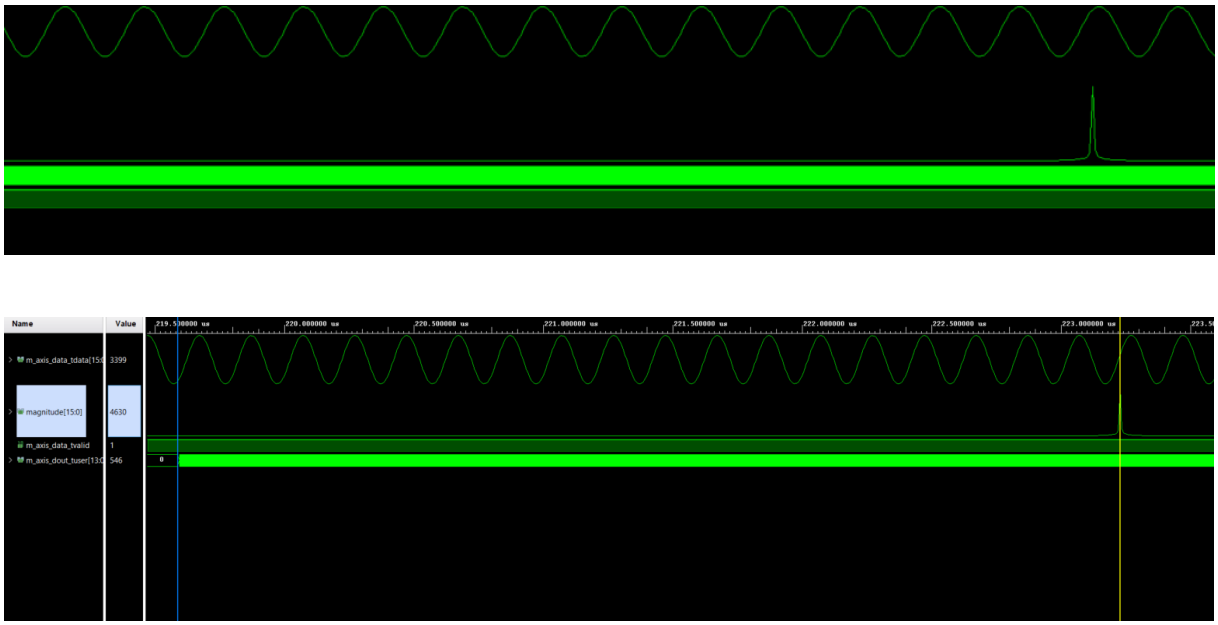


Figure 39: Chipscope Simulation

The sinusoidal signal emulated by the DDS Compiler is shown on the first line of the simulator output. The outcome of the computations conducted by the FFT and CORDIC IP Cores, which represents the frequency spectrum of the input signal from the DDS Compiler, can be seen in the second line. Because it is a sinusoidal signal, the frequency content of that signal is a single peak. The first step in calculating this signal's peak bin[2] number is to determine the frequency resolution. This may be accomplished by using equation 3.1. The second step is to use the resolution bandwidth value to determine where the frequency of any signal is situated by dividing the frequency by the resolution bandwidth, as stated in equation 4.2.

$$Bin_{number} = \frac{F_{signal}}{RBw} \tag{4.2}$$

Using the previous equations on the simulated system, the resolution bandwidth is 9.155 kHz. Because the sinusoidal signal has a frequency of 5 MHz, the bin number corresponding to this frequency is 546 when applying equation 4.2. After performing these calculations and examining the simulation results, the expected results match the simulation results, indicating that the implemented system is working correctly.

Another approach to compute the signal's amplitude is to use the CORDIC IP Core to calculate the absolute value of the FFT IP Core output. To accomplish this, the FFT IP Core output must be separated into the real and imaginary components. Applying these components on equation 4.3, the magnitude is calculated.

$$Magnitude = \sqrt{real^2 + imaginary^2} \tag{4.3}$$

To perform this computation, a multiplication IP Core for the real and imaginary component squares and a sum IP Core are required, as illustrated in Figure 40. After it, the CORDIC IP Core must be set to compute the square root.

[2] A frequency bin denotes a segment of the frequency axis

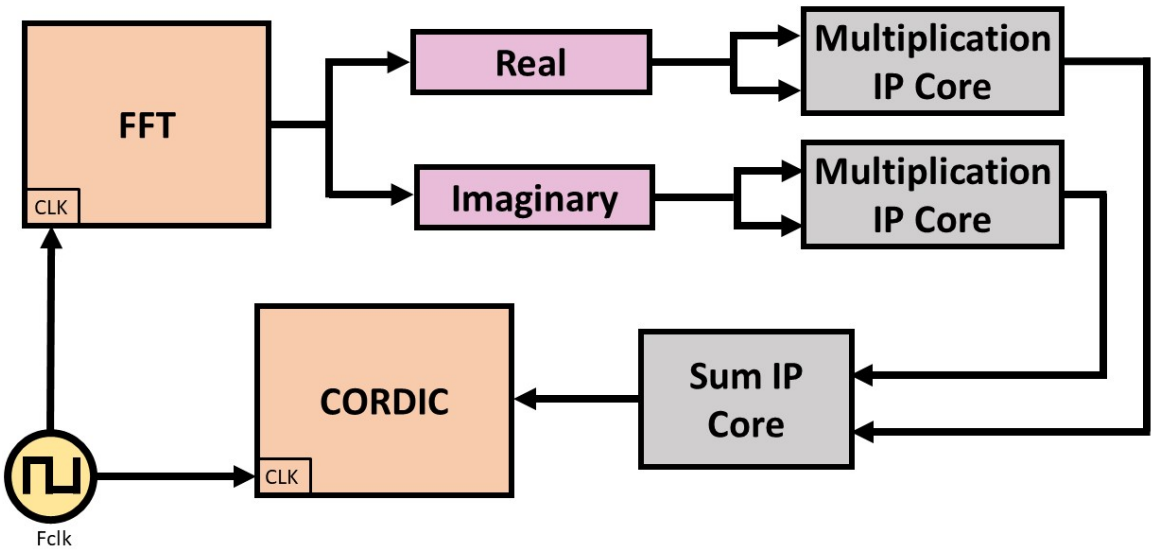


Figure 40: Magnitude Implementation

The issue with this solution is that the CORDIC IP Core cannot manage the sum IP Core output width while using the Unscaled setting on the FFT IP Core, because the width of the resultant data of the multiplication and sum would be larger than the maximum data width input of the CORDIC IP Core. Another reason not to use this approach is because vector translation on CORDIC IP Core is significantly easier and less resource consuming. Following these issues, and because this isn't the best approach, the implementation represented in Figure 38 was used, with the CORDIC IP Core set to vector translation.

4.2.3 Heterodyne architecture

Due to the hardware's limitations, an heterodyne architecture was implemented with the goal of increasing the frequency resolution of the spectrum analyzer by slicing the captured spectrum.

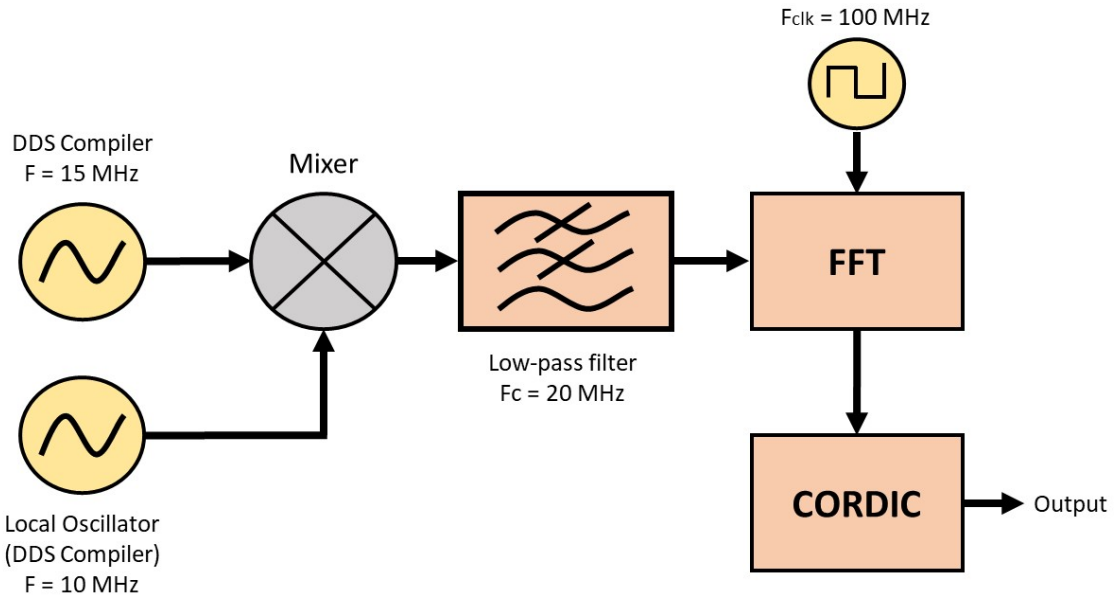


Figure 41: Heterodyne architecture simulation

The block diagram depicted in Figure 41 was used to simulate this architecture. The DDS Compiler was used for the main signal and the local oscillator, with frequencies of 15 MHz and 10 MHz, respectively. FIR Compiler IP Core is used for the low-pass filter. It has a 32-bit output and a single-rate type because single-rate digital filters do not change the sampling frequency of a signal during the filtering process. As a result, when a single-rate filter is applied to the input signal, the output signal has the same sampling frequency as the input signal. The output should have as many bits as possible to have the highest precision, but 32 bits for each component were used to fit the previous design because the FFT IP Core maximum bits for each component is 34 bits and these two extra bits are irrelevant in this case in terms of precision. As a result, the FFT IP Core input width has to be increased to 32 bits for each component. FIR Compiler IP Core implements a FIR Filter using coefficients, which were calculated using a filter designer [30]. Figure 42 depicts its frequency response.

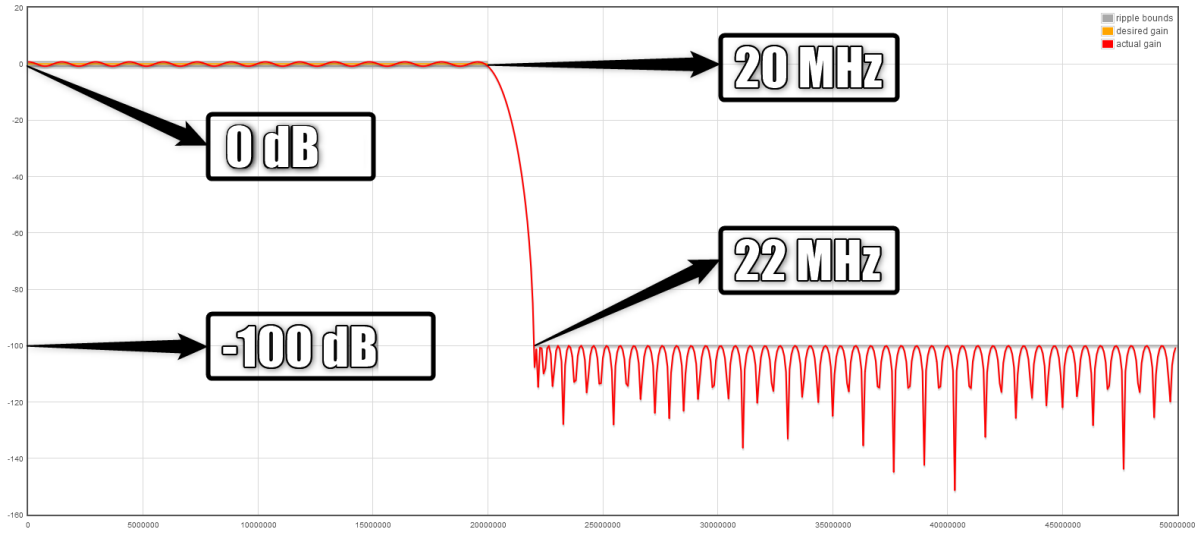


Figure 42: Filter Characteristics

The low-pass filter, as shown in Figure 42, has a cut-off frequency of 20 MHz and its window is set to rectangular, which was chosen mainly because it is a good choice for viewing transient signals, although it is less useful than the others as a general-purpose window[31]. The filter has a 2 MHz transition bandwidth, meaning that it will cut completely frequencies above 22 MHz. When the two signals are mixed, the main signal's frequency (15 MHz) is going to shift 10 MHz (local oscillator's frequency) to the right, therefore, the output signal of the mixer should have signal components on 5 MHz and on 25 MHz. The component of 25 MHz is going to be filtered by the low-pass filter described before.

The FFT IP Core was configured with 1024 points for simulation purposes, as the simulation would run faster. Using the equations mentioned in 3.1 and 4.2, the expected result is a peak on bin 51, as calculated on Equation 4.4.

$$1024 \times \frac{(15MHz - 10MHz)}{100MHz} \approx 51 \tag{4.4}$$

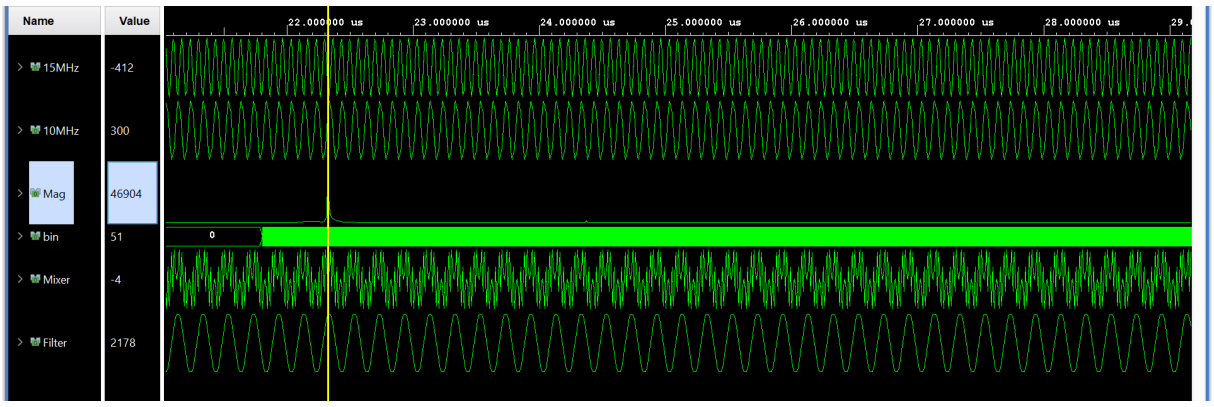


Figure 43: Heterodyne Chipscope Simulation

In the simulation results depicted in Figure 43, five signals are shown respectively:

- 15 MHz local oscillator;
- 10 MHz main signal;
- Spectrum;
- Mixer's output signal;
- Filtered signal;

The bin number is also displayed in the fourth line to ensure that the results are accurate. Analyzing the simulation results, the first bin number corresponds to 10 MHz and bin 51 corresponds to 15 MHz, indicating that the results are as expected. This architecture was implemented on the Zedboard after it was simulated.

4.3 FPGA synthesis and implementation

4.3.1 Spectrum calculation

After the validation of the architecture and prototype implementation by simulations, the DDS compiler and the System Clock Simulation were replaced by the AD9467 IP Core from Analog Devices, where the system clock is sourced directly to the evaluation board and the input signal is given by the ADC acquisition. The resulting block diagram responsible for the signal capture and DSP spectrum calculation is shown in Figure 44.

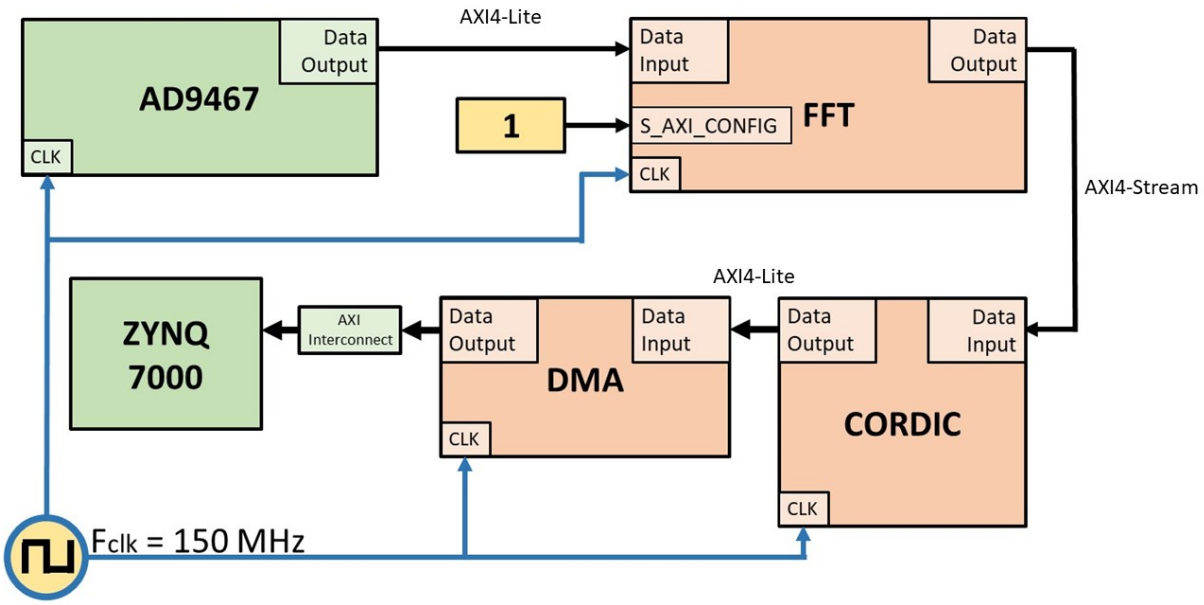


Figure 44: AD9467 spectrum calculation block diagram

To test the system, two function generators were used in the test setup, one for the signal input and the other for the clock. Figure 45 represents the test setup.

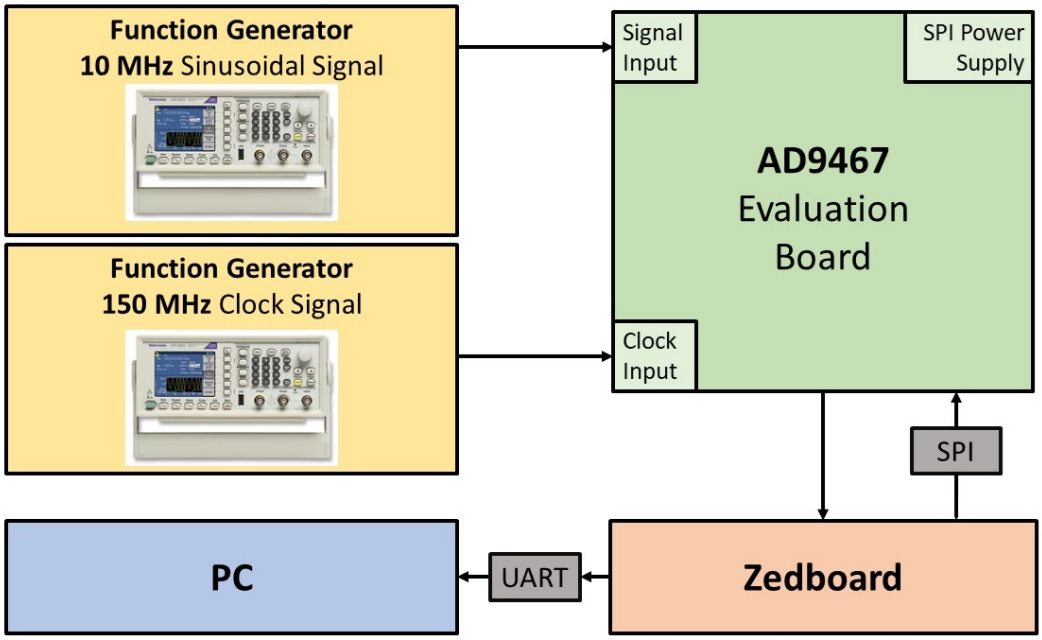


Figure 45: Block diagram of the test setup for the acquisition system and spectrum calculation test

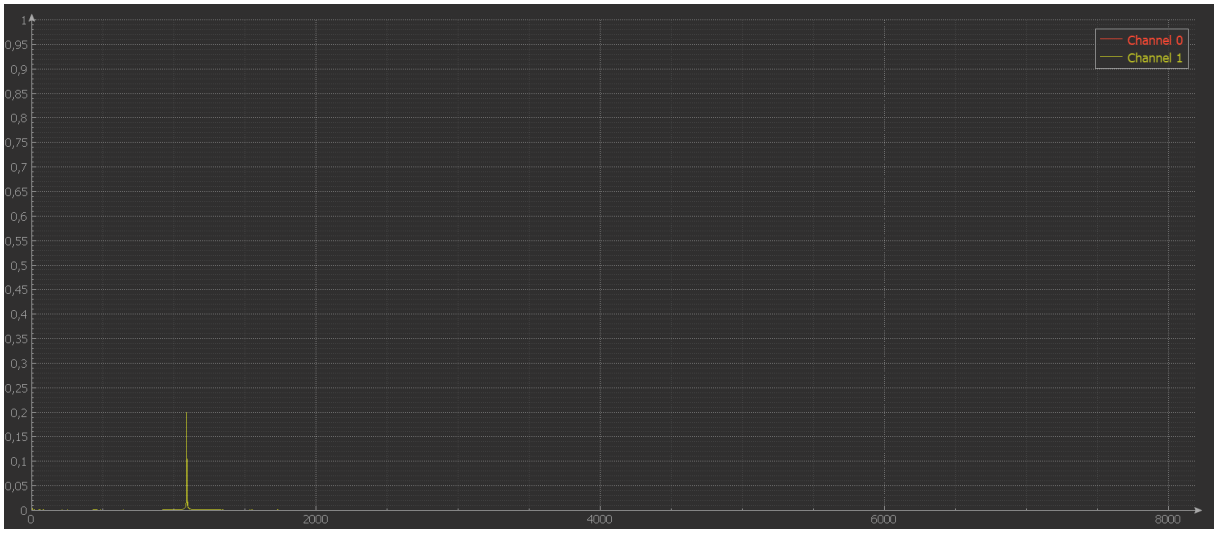
Using the ILA, the output from the CORDIC IP Core could be analyzed to see if it corresponded to the expected results. Three output signals were connected to the ILA in order to analyze the output: AD9467, CORDIC, and the tuser from the FFT IP Core. The NFFT configured in the FFT IP Core corresponds to the *tuser* output, which will show the frequencies bin numbers. Before analyzing the results, a few calculations were performed to ensure that the calculated results matched the system output. Equations 3.1 and 4.2 were used to validate the results with a 1 MHz sinusoidal signal, 16384 FFT points, and a 150 MHz clock. The peak on the signal's fundamental frequency, accordingly, should appear on bin number 109, and as expected, it was located in bin number 109.

After analyzing and validating the results on Chipscope, the next step was to test and verify the system results with the software component, due to the importance of the configurations made to the AD9467.

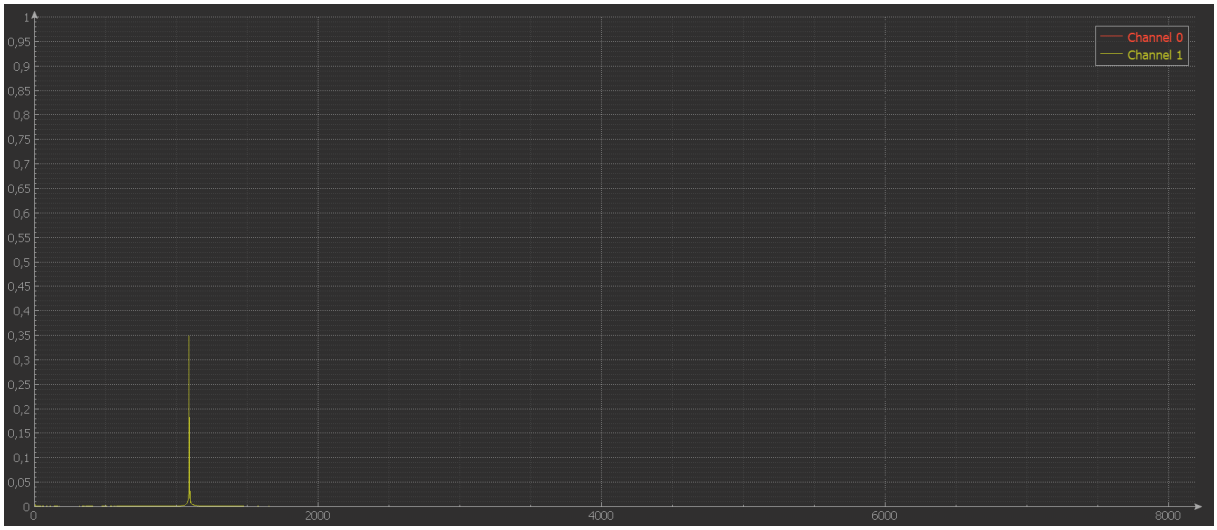
To test the system with the software component, a few modifications were made to the equation that translates the digital values to real values. The output width from CORDIC IP Core is 16 bits, but according to the CORDIC IP Core datasheet, the output bit width changed to 13 bits due to the output QNUMBER format. Hence, the voltage of the output signal can be calculated by changing N_{bit} in Equation 4.1 to 13.

$$CORDIC_{Amplitude} = CORDIC_{RawAmplitude} \times \frac{2,5V}{2^{13}} \tag{4.5}$$

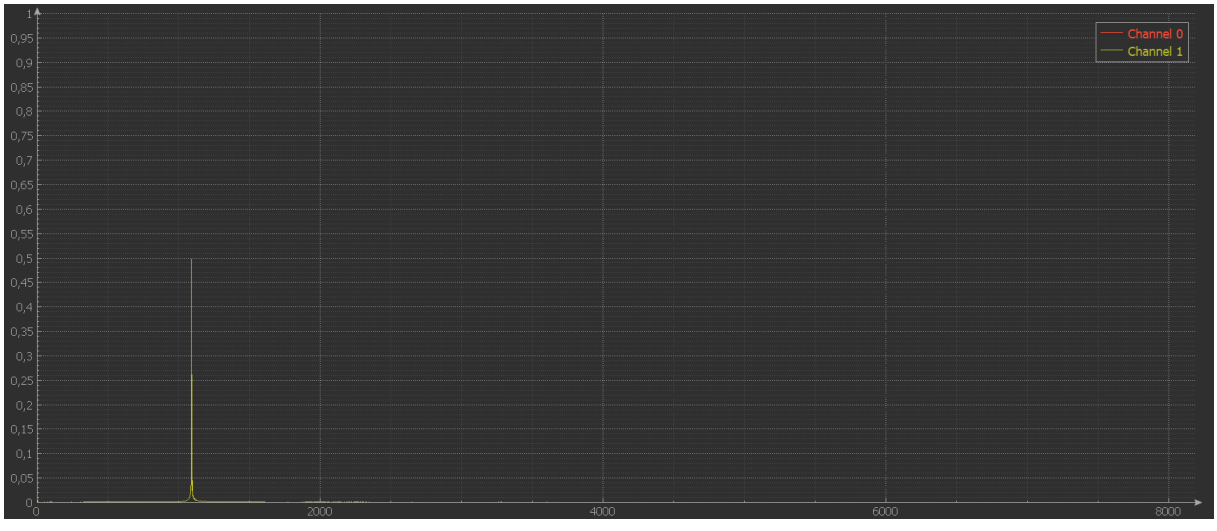
To validate the system, sinusoidal signals of varying amplitudes were fed into the ADC. The external clock was set to a frequency of 150 MHz. Figure 46 depicts the results.



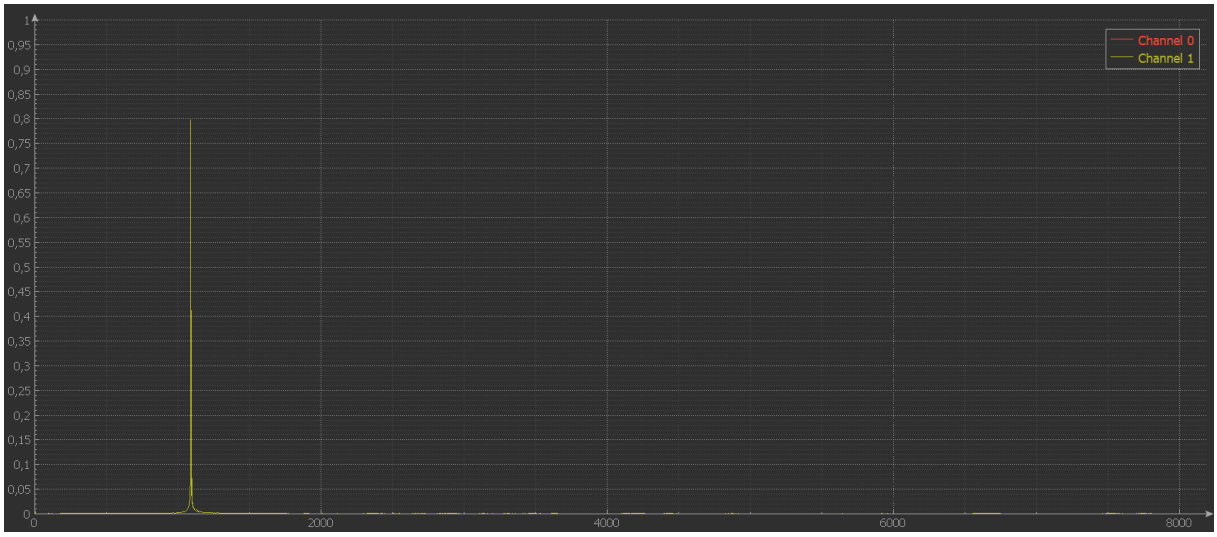
(a) Software test 200 mV



(b) Software test 350 mV



(c) Software test 500 mV



(d) Software test 800 mV

Figure 46: Software Tests

When the test results are analyzed, the output values from each test match the expected results, with the bin number matching its frequency and the amplitude matching the amplitude of the function generator's output signal. A spectrum analyzer typically displays the spectrum in terms of power such as dBm (decibels-miliwatt) or $dB\mu V$ (decibels-microVolt), therefore, a conversion between voltage and signal power had to be calculated. The equations 4.6, 4.7 and 4.8, were respectively implemented on the software component in order to do the conversion, in this case, to dBm.

$$V_{rms} = \frac{CORDIC_{Amplitude}}{2} \times \frac{\sqrt{2}}{2} \tag{4.6}$$

$$P_V = \frac{V_{RMS}^2}{Z_0} \tag{4.7}$$

$$P_{dBm} = 10 \times log_{10}(P_V \times 1000) \tag{4.8}$$

To convert voltage to power levels, one needs to calculate the Root Mean Square (RMS) value of the signal voltage and then convert to power having in consideration the system reference impedance (Z0) is 50 Ohm, as described in equations 4.7 and 4.8. With this new implementation, the software component was tested again.

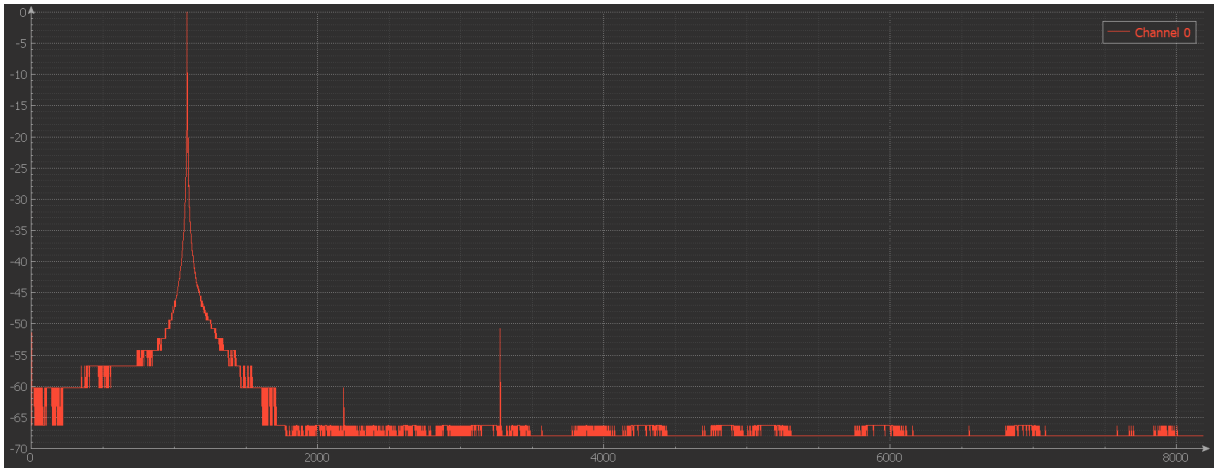
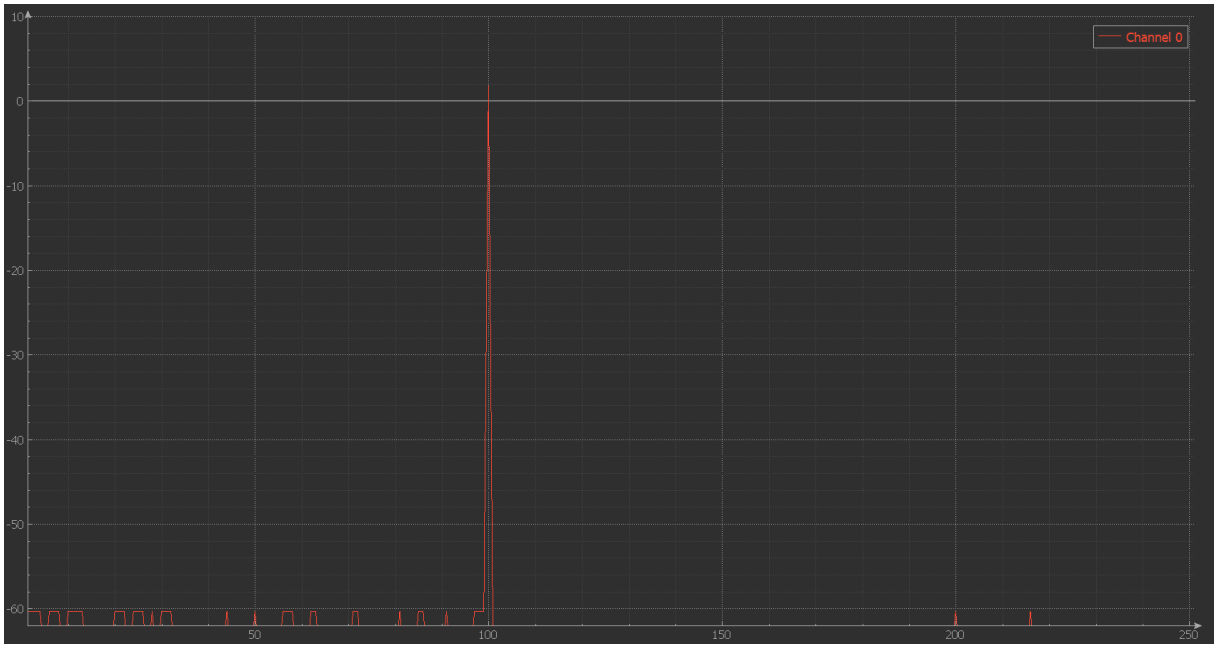


Figure 47: Software test dBm

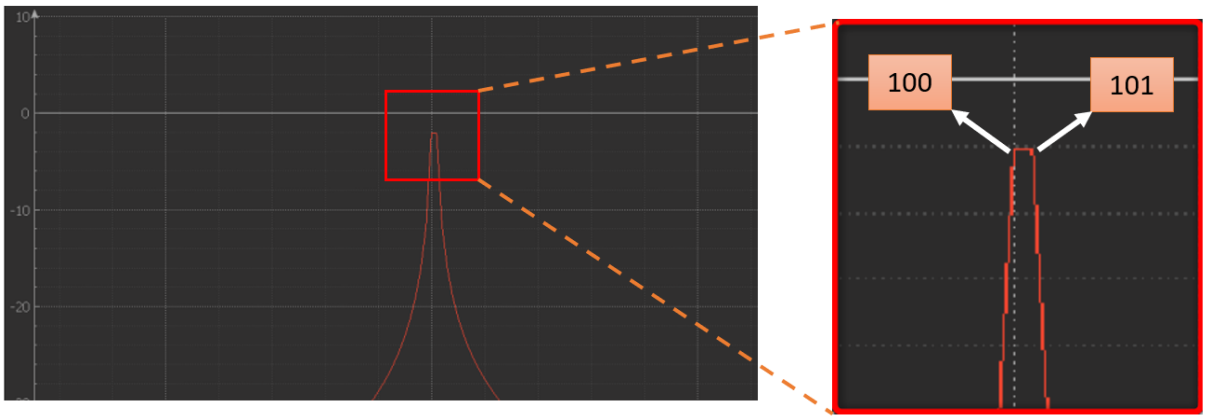
Using equations 3.1 and 4.2, and with a sinusoidal signal with 10 MHz and 0 dBm, the bin number correspondent to this frequency is 1092. This input signal's first and second harmonics, due to the signal being digital, appear as well on the spectrum in the bin 2184 and 3276 respectively. Figure 47 depicts the test results, which are consistent with the predicted results. The system is now able to acquire signals and calculate its spectrum.

4.3.2 System requirements and limitations

To verify the resolution bandwidth, a 1.5258 MHz signal was used in order to the frequency peak appear on bin 100, according to equations 3.1 and 4.2. Consequently, the bin number 101 corresponds to 1.5411 MHz. According to these calculations, if a signal with a frequency of 1.5335 MHz is fed to the AD9467, as it is out of the resolution bandwidth, a peak is expected to appear on bin 100 and 101. Figures 48a and 48b demonstrate the findings of this test, which were as expected.



(a) Resolution Bandwidth Test - Frequency in the RBW



(b) Resolution Bandwidth Test - Frequency out of the RBW

Figure 48: Resolution Bandwidth Test

In order to increase it, the solution is either changing the clock frequency or increasing the number of FFT points. As it is impossible to increase the number of points due to the hardware's limitations, the solution is to decrease the clock frequency. But then, given the Nyquist theorem, this would reduce the spectrum span, and since there's also a requirement to have a span of 108 MHz, this cannot be compromised either. This is where the implementation of the heterodyne architecture enters to overcome this problem.

4.3.3 Heterodyne architecture

As previously stated, the FFT clock must be reduced in order to increase the spectrum analyzer's resolution bandwidth. In order to get a resolution bandwidth of 2 kHz, the total bandwidth of the spectrum must be divided into eight bandwidths of 16.384 MHz by using the DDS Compiler as local oscillator. As for the clock, using the equation on 3.1, the clock required is 32.768 MHz.

According to Nyquist's theorem, the FFT IP core must be clocked with twice the spectral bandwidth to calculate the spectral bandwidth of 16.384 MHz, as depicted in Figure 49

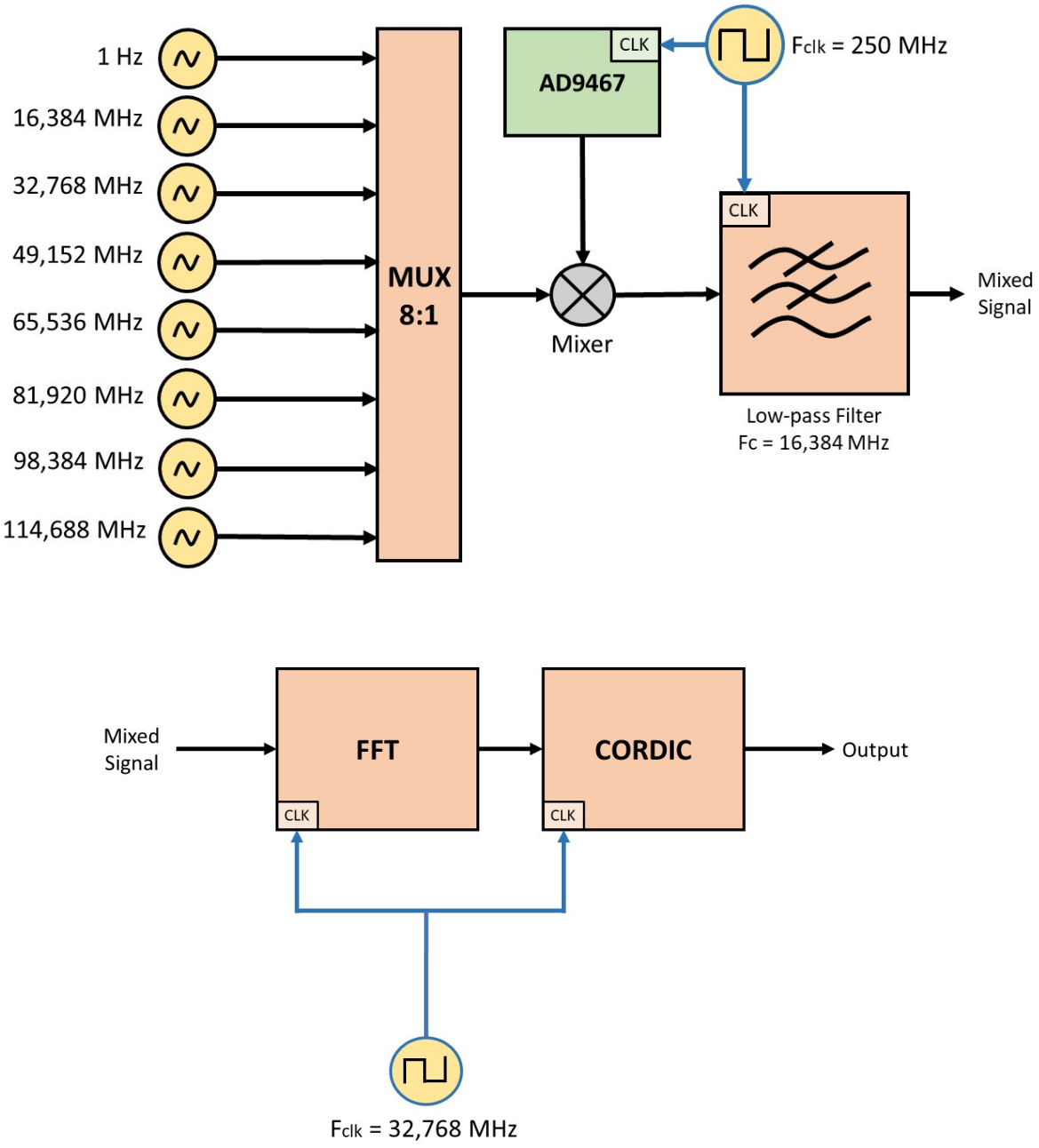


Figure 49: Heterodyne architecture

After this analysis, the final block diagram is implemented as depicted in Figure 50, containing all the IP Cores and modules required to build the final system.

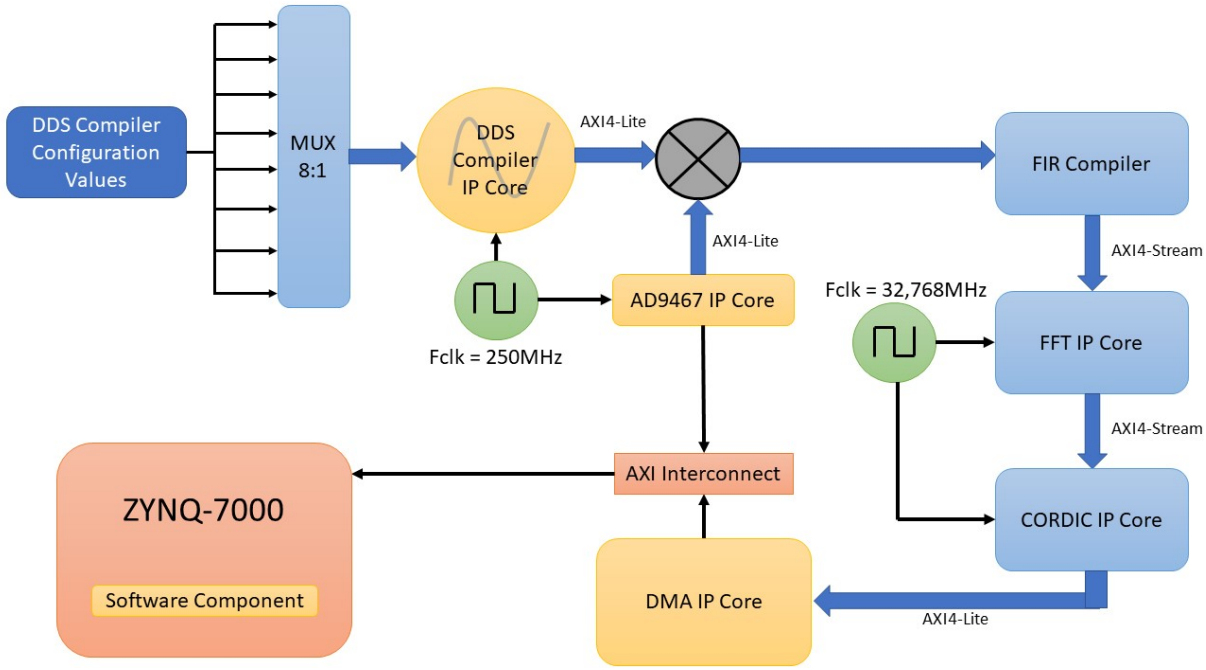


Figure 50: Final System Block Diagram

Regarding the filter, a new set of coefficients were calculated in order to fit in the designed system. The cut-off frequency for this filter was set to 16.384 MHz, and its window was set to rectangular for the same reason as the filter used in the simulation. The cut-band attenuation was limited to -35 dB. The reason for this limitation is explained further in the following subchapter. The filter characteristics are depicted in Figure 51.

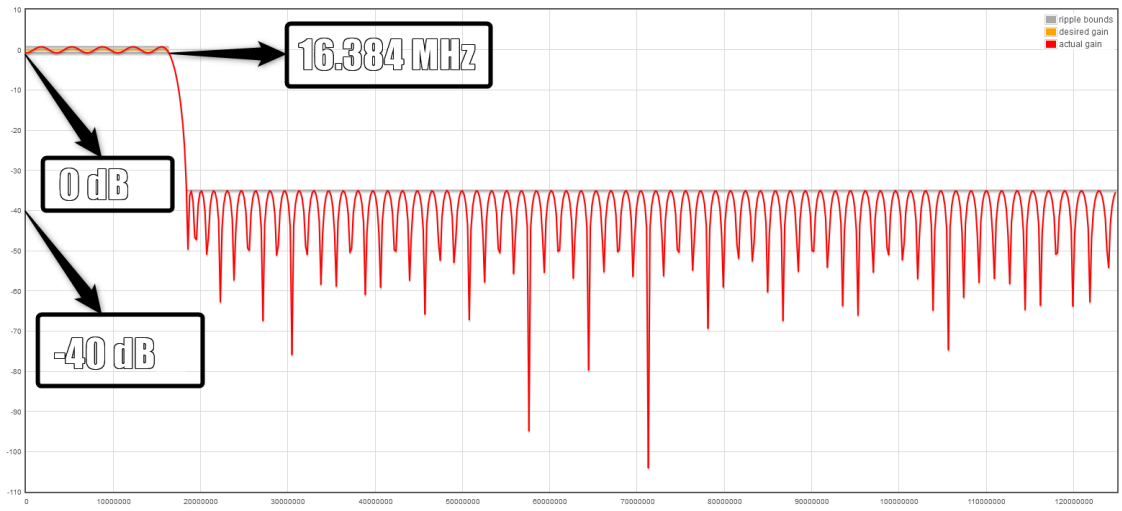
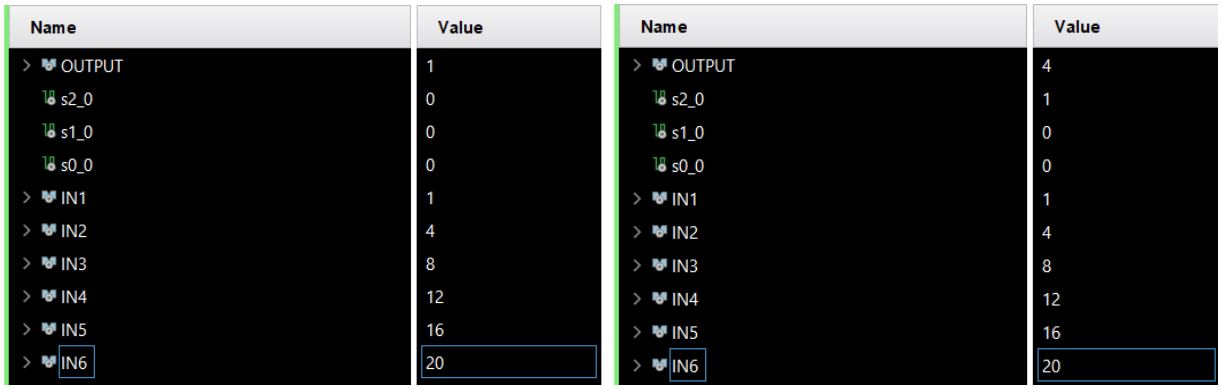


Figure 51: Filter implementation for heterodyne system

To select the spectrum analyzer's band of analysis, a multiplexer for the local oscillator was implemented. The multiplexer's inputs are built-in switches on the Zedboard. Before the multiplexer was added to the system, a test was run to validate its behavior. To put it to the test, the possible outputs were set to 1, 4, 8, 12, 16, and 20. Figure 52 shows two examples of the results of these tests.



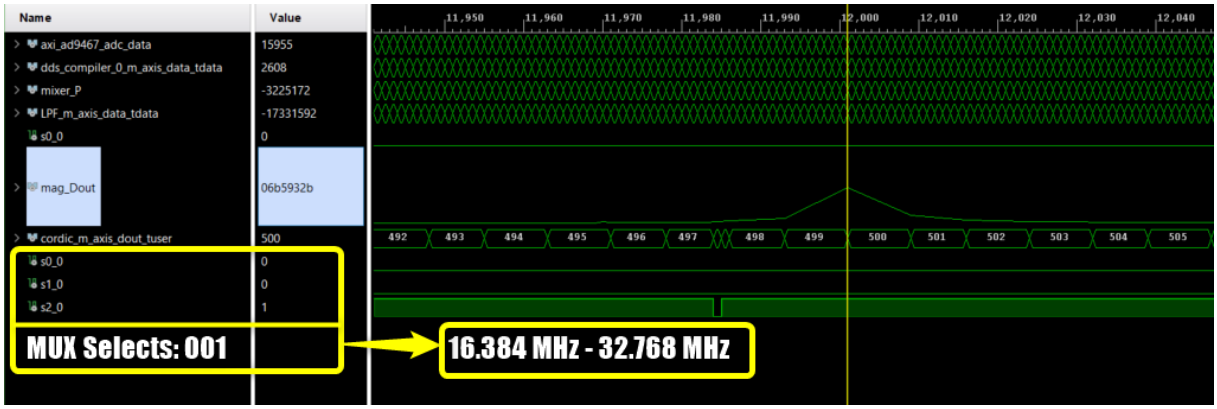
(a) MUX First Output (b) MUX Second Output

Figure 52: MUX Validation

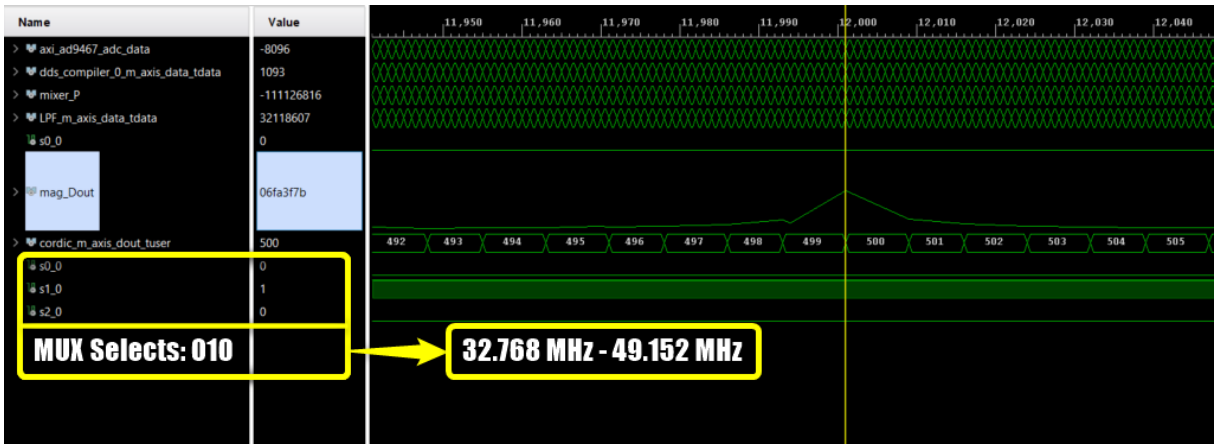
Following the validation of these tests, the multiplexer output values were changed. A DDS Compiler with configurable phase increment was used as a local oscillator. Using the multiplexer with outputs set to configure the phase increment, the desired frequency can be communicated to the DDS Compiler IP Core.

According to equations 3.1 and 4.2, using a 1 MHz input signal, the peak should appear at bin 500. The Chipscope was used to validate the results. If the input signal is always 1 MHz above the local oscillator frequency, shifting the local oscillator should always result in the same frequency peak in the same bin.

The chipscope results are as expected, with the peak appearing in bin 500 in every spectrum analyzer bandwidth selected. For the tests in the bandwidth between 16.384 MHz and 32.768 MHz, a signal of 17.384 MHZ was fed to the AD9467 input. These results are depicted in 53a. The results for the bandwidth range of 32.768 to 49.152 MHz are shown in 55.



(a) Heterodyne Architecture AD9467 Chipscope Test [16.384 MHZ - 32.768 MHZ]



(b) Heterodyne Architecture AD9467 Chipscope Test [32.768 MHZ - 49.152 MHZ]

Figure 53: Heterodyne Architecture AD9467 Chipscope Test

Using a bandwidth of 16.384 MHz to 32.768 MHz and an input signal of 18 MHZ, the expected bin number is 808.

$$16384 \times \frac{(18MHz - 16.384MHz)}{250MHz} = 808$$

The results shown in 54 correspond to the expected results.

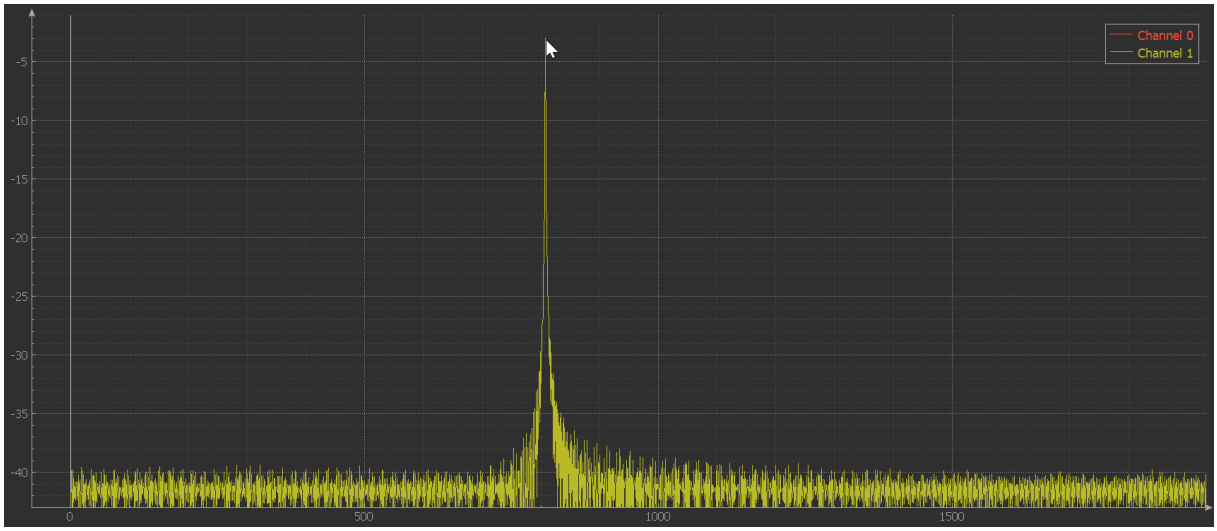


Figure 54: GUI test 18 MHZ input with [16.384 - 32.768 MHZ]

After the results have been validated on the GUI, the intended resolution bandwidth of 2 kHz must be validated. Another test was performed to validate this. Bin 809 should correspond to the frequency 18.2 MHz if bin 808 corresponds to the frequency 18 MHz.

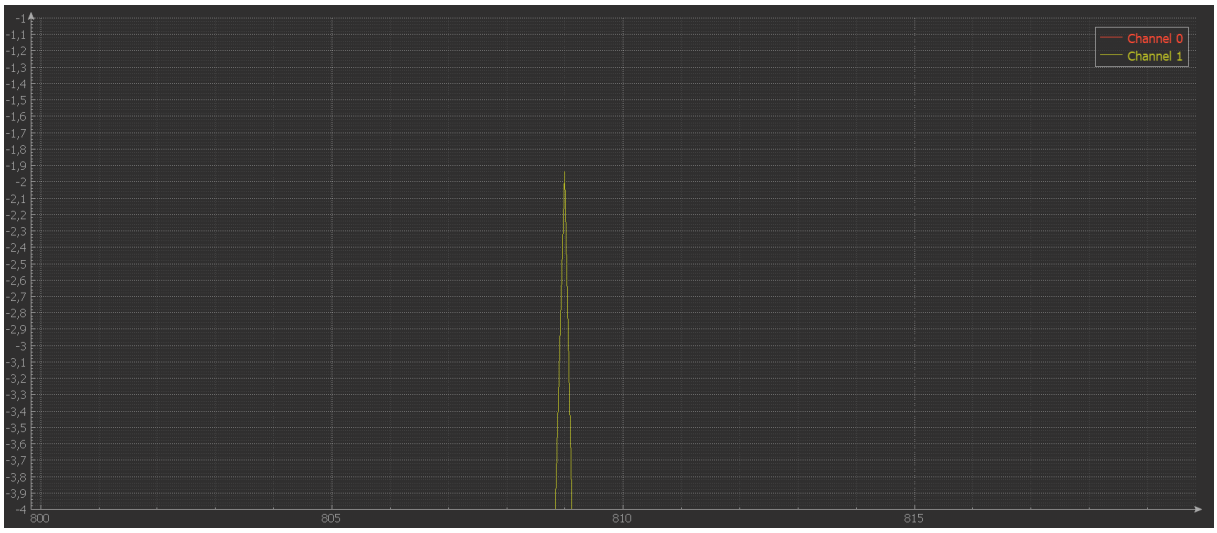
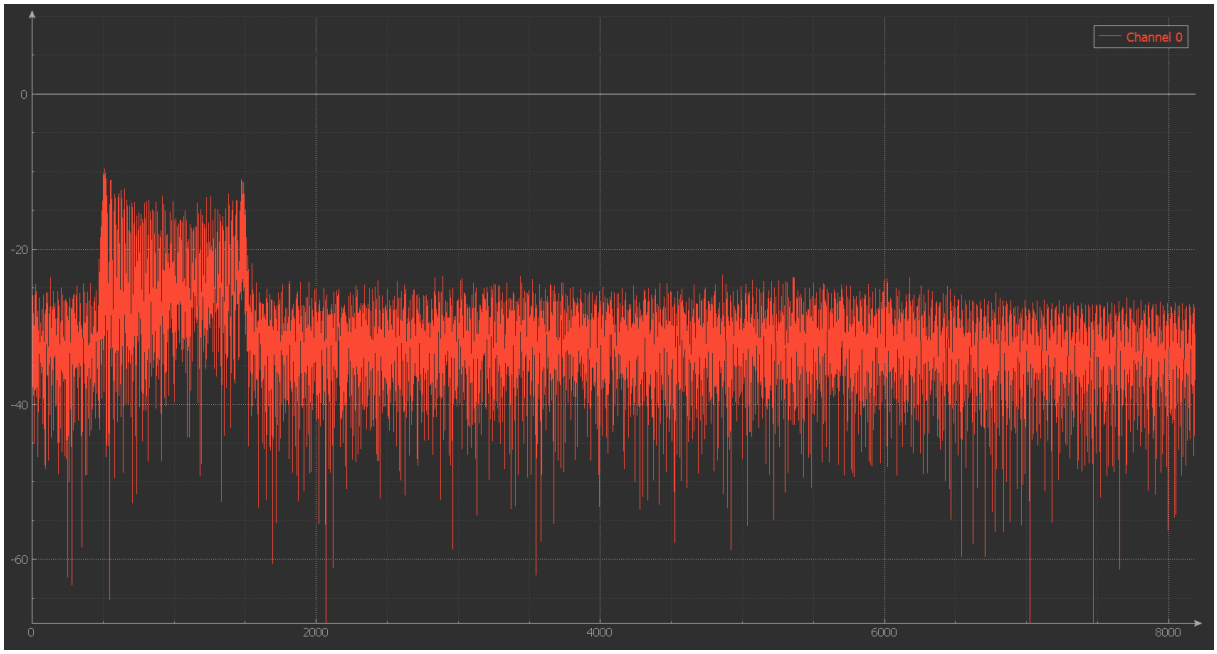


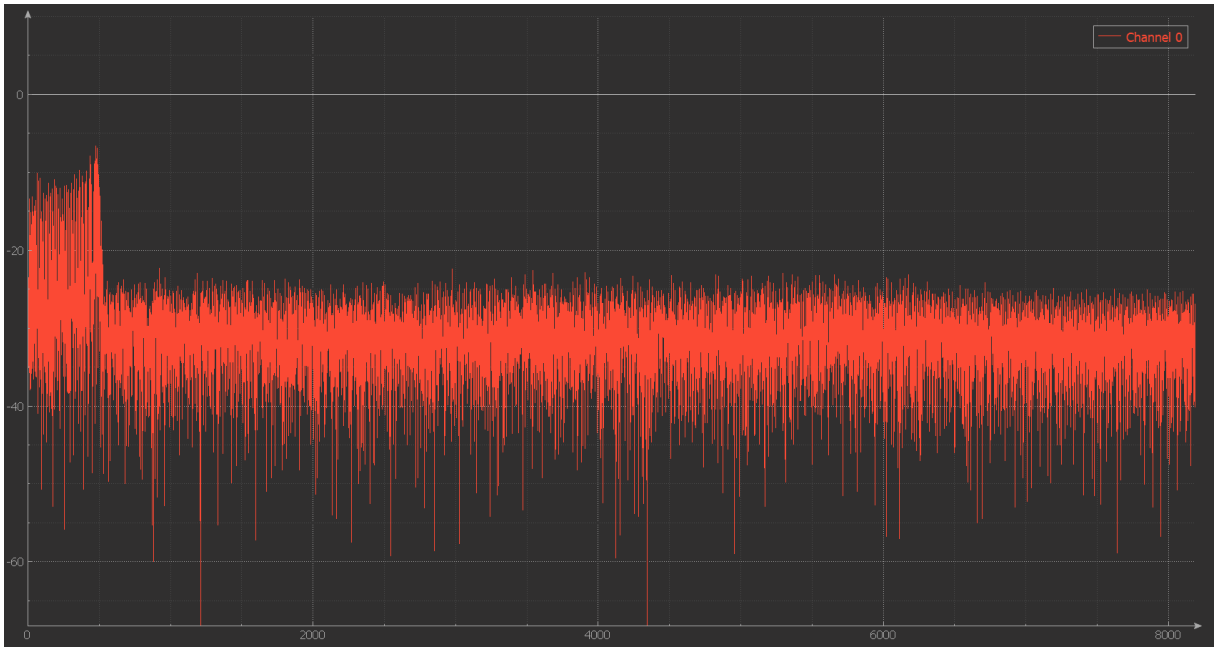
Figure 55: Resolution bandwidth Validation - 18.2 MHz

The results depicted in Figure 55 match the expected results, validating the spectrum analyzer's 2 kHz resolution bandwidth.

Finally, to validate the 16.384 MHz bandwidth, two frequency modulated signals (FM signals) with a 1 Mhz deviation were fed to the ADC, one with 16.384 MHz and the other with 18.384 MHz.



(a) Bandwidth Validation - FM Signal with 18.384 MHz Carrier and 1 MHz Deviation



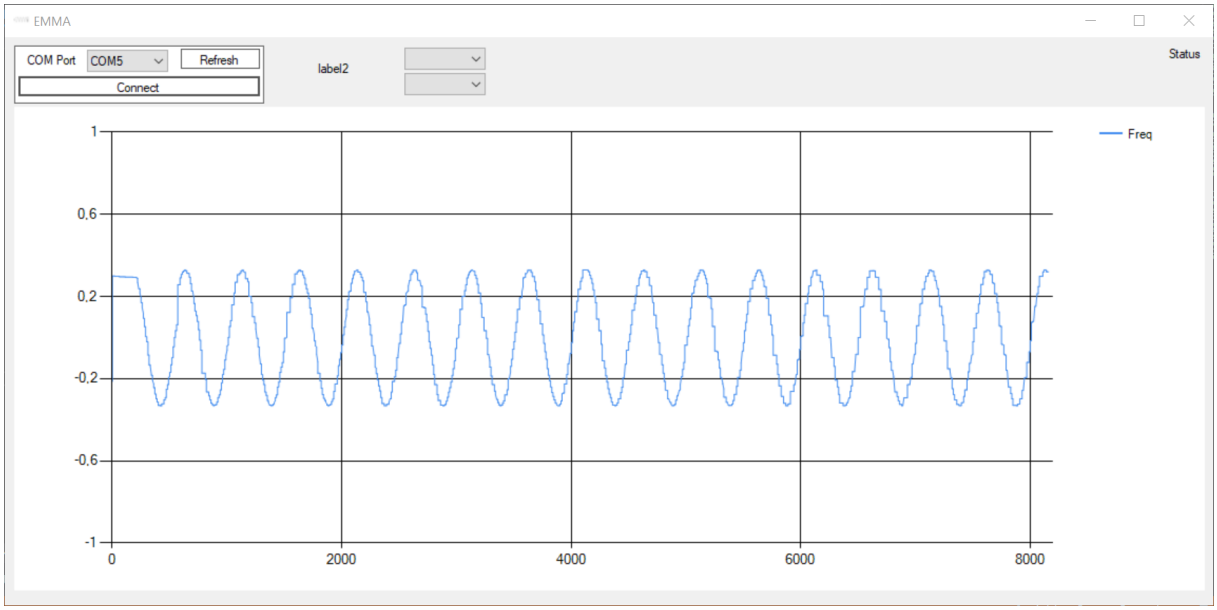
(b) Bandwidth Validation - FM Signal with 16.384 MHz Carrier and 1 MHz Deviation

Figure 56: Bandwidth Validation Test with FM Signal

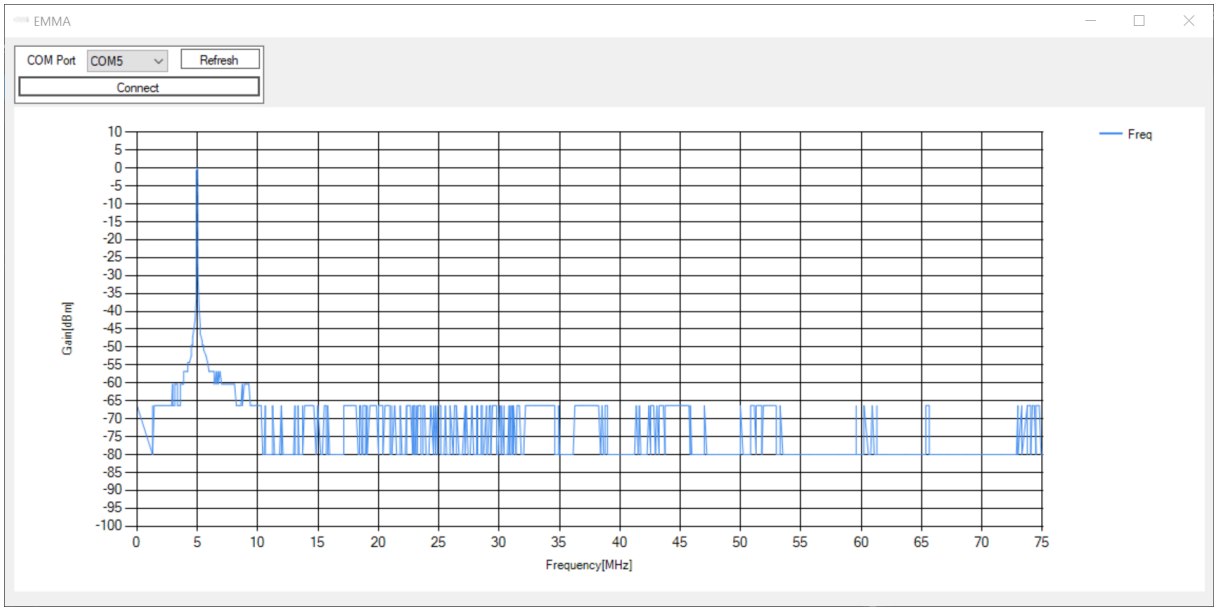
The FM signal, as shown in 56a, has a carrier frequency of 18.384 MHz and a deviation of 1 MHz, indicating that it contains frequency components to its left and right. In the case of an FM signal with a carrier frequency of 16.384 MHz, as shown in Figure 56b, the frequencies on the right side of the carrier frequency are the only ones that appear, implying that the frequencies on the left side don't appear since it falls inside the bandwidth of 1 Hz to 16.384 MHz. The results shown in Figure 56 validation are consistent with what was predicted.

4.3.4 Graphics User Interface

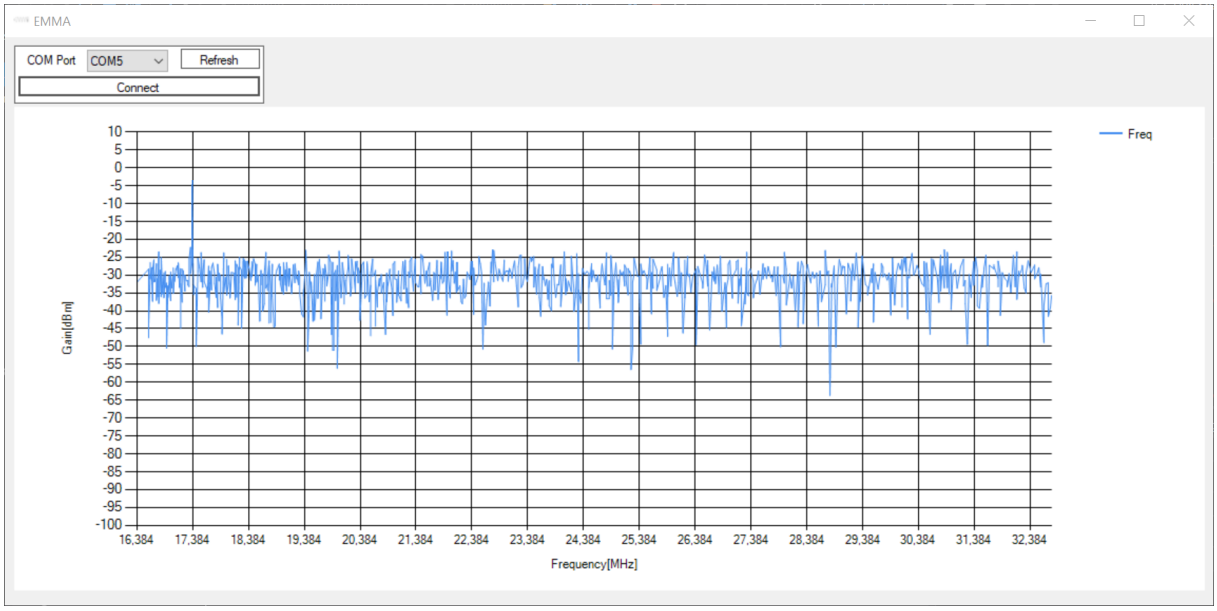
A GUI was created after confirming the spectrum analyzer's correct operation. Instead of bin numbers, the X axis displays the proper frequencies. Figure 57 demonstrates signal acquisition, basic spectrum acquisition, and spectrum acquisition with a heterodyne architecture using this GUI. In order to confirm the correct operation of this GUI, multiple settings on the input signal were employed for these experiments.



(a) Signal Acquisition GUI Test



(b) Spectrum Acquisition GUI Test



(c) Heterodyne Architecture GUI Test

Figure 57: Tests to the developed GUI

A sinusoidal signal with a frequency of 5 MHz was fed to the AD9467 for the signal acquisition test. Figure 57a depicts the results. For the non-heterodyne spectrum analyzer, a 5 MHz sinusoidal signal with 0 dBm was provided to the AD9467. The test results are depicted in 57b, with the X axis scaled in MHz and the Y axis scaled in dBm. For the heterodyne spectrum analyzer, a signal of 17.384 MHz and -5 dBm was sent to the AD9467, and the span chosen for this test is between 16.384 MHz and 32.768 MHz. The findings are as predicted, as shown in 57c, with a peak on 17.384 MHz with an amplitude of -5 dBm.

Chapter 5: Tests and results

5.1 Spectrum analyzer's noise floor

Concerning the noise floor, the final version of the spectrum analyzer does not meet the initial requirements, and many factors contributed to this problem, which will be discussed in this section. Starting with the AD9467 Evaluation board, it has 16 bits ($Nbit = 14$ active bits) and reads to a maximum of $V_{ref} = 2.5V$, meaning that the noise floor has a theoretical value of approximately -74 dBm. In order to calculate this value, the minimum value that he ADC can read must be calculated using equation 5.1.

Beginning with the AD9467 Evaluation board, it contains 16 bits ($Nbit = 14$ active bits) and reads to a maximum of $V_{ref} = 2.5V$, implying a theoretical noise floor of approximately -74 dBm. To determine this noise floor value, the lowest value that the ADC can read must be calculated using equation 5.1.

$$V_{ADCmin} = \frac{V_{ref}}{2^{Nbit}} \approx 1.22 \times 10^{-4} \tag{5.1}$$

Using equations 4.6, 4.7 and 4.8, the noise floor of the ADC can be calculated, as depicted in 5.2, 5.3 and 5.4.

$$V_{rms} = \frac{V_{ADCmin}}{2} \times \frac{\sqrt{2}}{2} \approx 4.32 \times 10^{-5} \tag{5.2}$$

$$P_V = \frac{V_{RMS}^2}{Z_0} \approx 3.73 \times 10^{-11} \tag{5.3}$$

$$P_{dBm} = 10 \times log_{10}(P_V \times 1000) \approx -74dBm \tag{5.4}$$

When compared to top-tier industrial spectrum analyzers, the top-tier industrial spectrum analyzers can reach a noise floor of around -90 dBm when used for this purpose. According to the equations, one solution to this problem is to use an ADC with a lower reading span or with more bits. For example, if the maximum reading is 1 V, the noise floor can reach -80 dBm.

5.2 System testing with DUT

A DUT Test was performed in order to compare a top-tier industrial spectrum analyzer. In order to be able to test the emissions on both spectrum analyzers at the same time, a signal splitter was used. A Rhode&Schwarz EMI receiver and the developed spectrum analyzer were used to capture the emissions of a DUT device. The Rhode&Schwarz EMI receiver settings and its results are depicted in Figure 58. Regarding the setup, the EMI Receiver was setup as a spectrum analyzer, with an RBW of 300 Hz, a VBW of 300 Hz and a bandwidth between 1 Hz and 16.384 MHz. With these settings, as mentioned above, the spectrum analyzer has a noise-floor around -90 dBm.

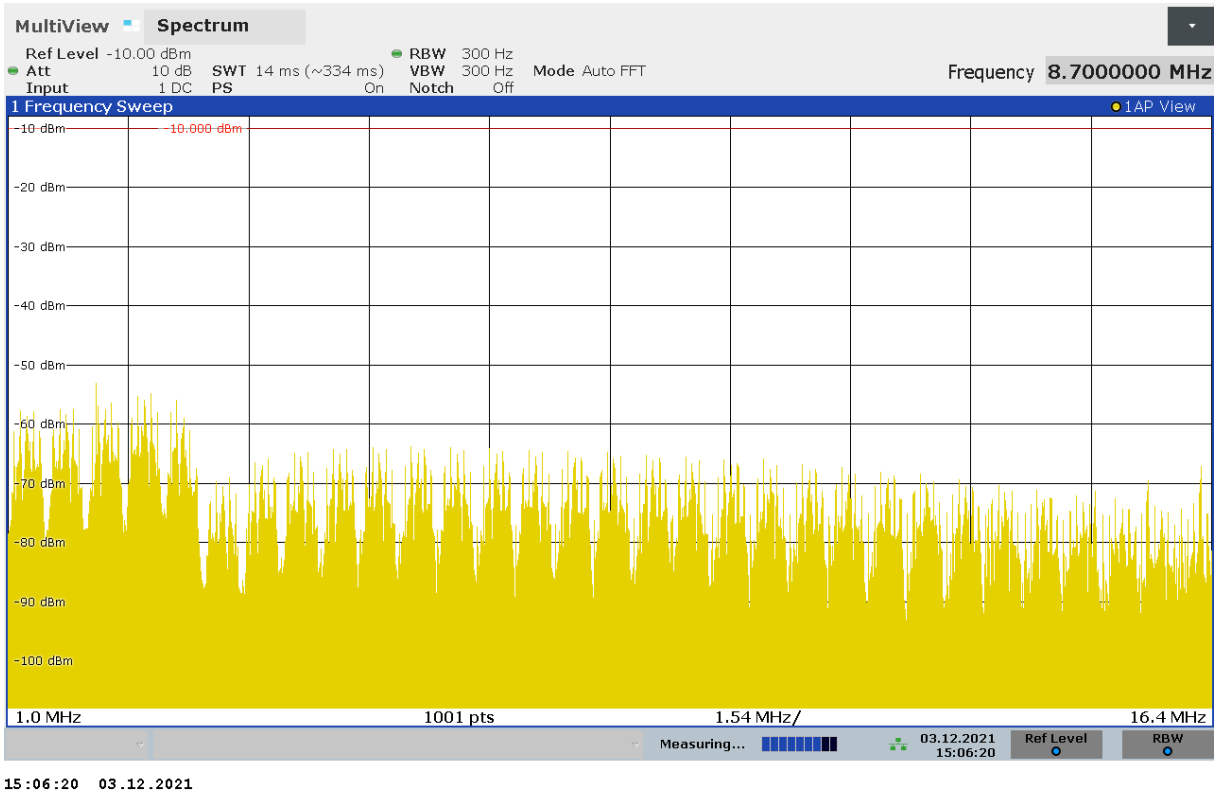


Figure 58: DUT Test Rhode&Schwarz EMI Receiver

Regarding the developed spectrum analyzer, two tests were performed: (1) a test with the non-heterodyne architecture and (2) a test with the heterodyne architecture. The spectrum analyzer with non-heterodyne architecture was able to pickup frequencies with an amplitude above -70 dBm. Comparing to the Rhode&Schwarz EMI receiver, this noise floor is not low enough, as some frequencies from conducted emissions have amplitudes lower than -70 dBm.

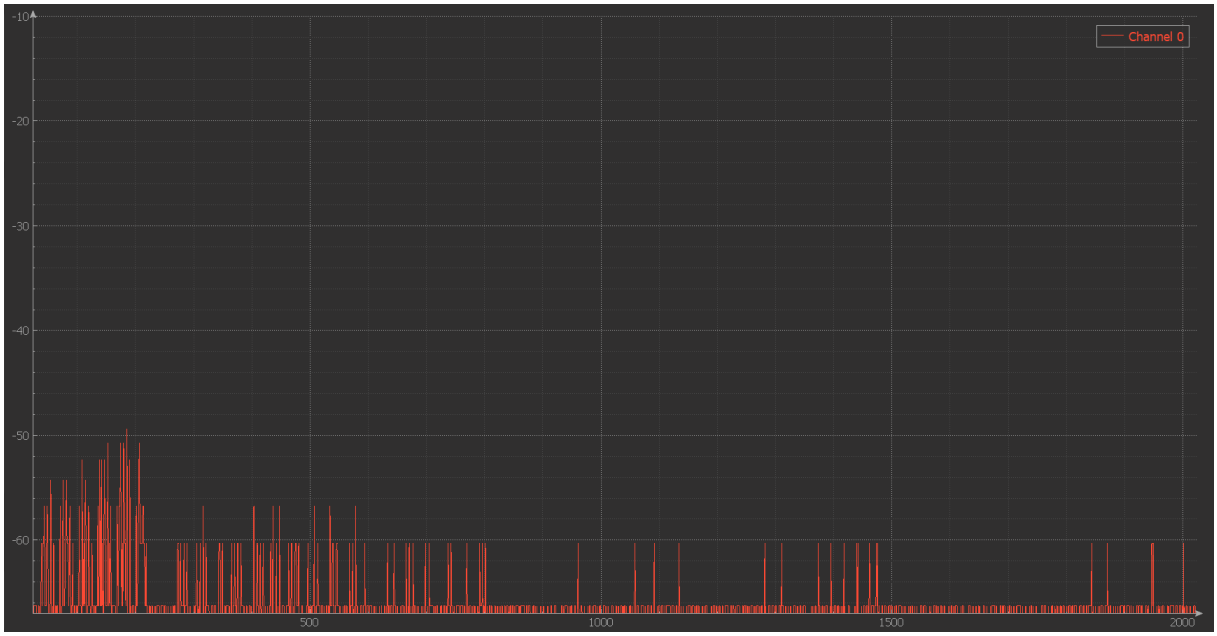


Figure 59: DUT Test with developed Spectrum Analyzer (non-heterodyne)

Because the solution was the implementation of a heterodyne architecture, the use of the DDS compiler and a filter also contributed to the noise floor failing to meet the requirements. When no input signal is present on the heterodyne system, unlike the non-heterodyne system, there is noise present rather than zeros. The noise floor is between -32 dBm and -40 dBm with no signal at the input, as shown in Figure 60.

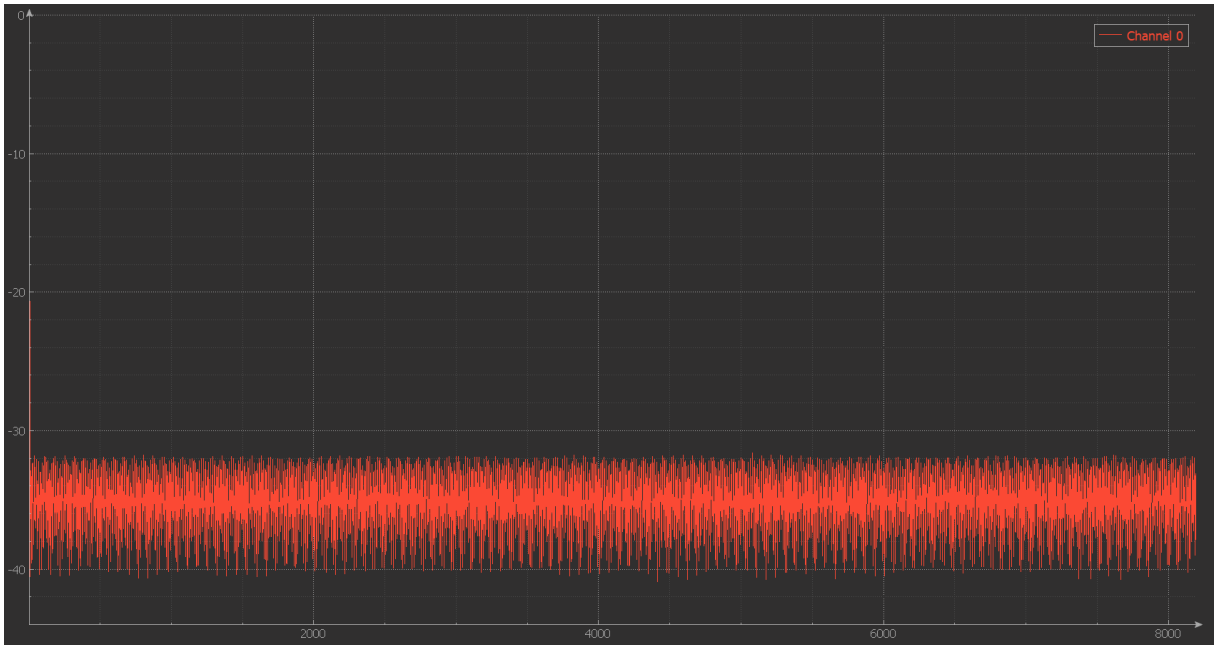
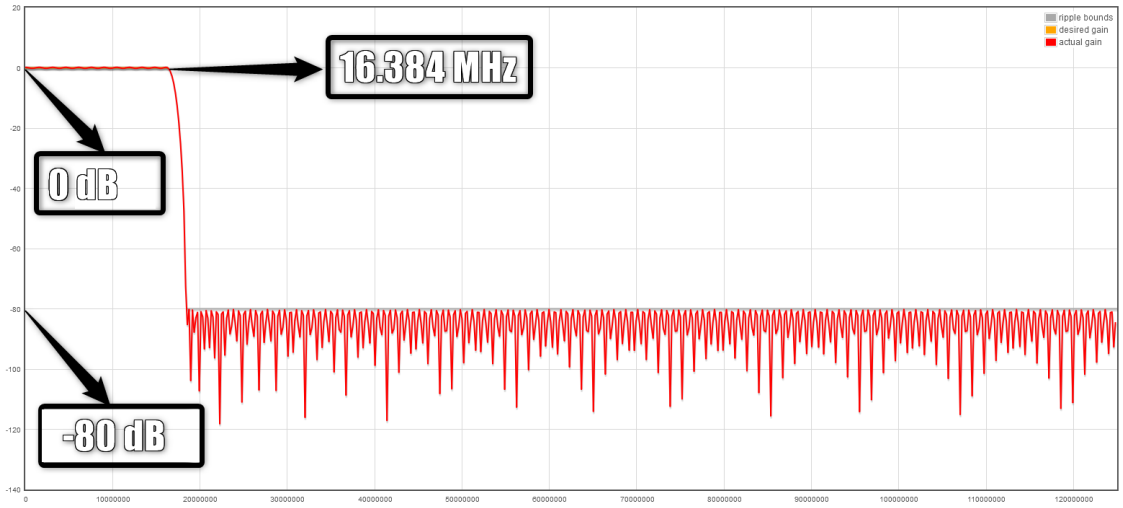


Figure 60: Noise Floor test on Spectrum Analyzer with Heterodyne Architecture

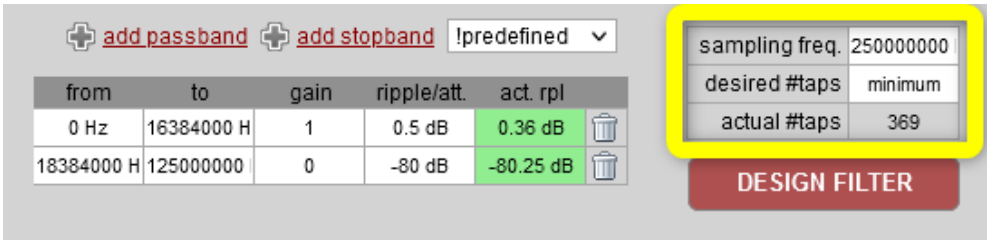
In comparison with the Rhode&Schwarz EMI receiver, the spectrum analyzer with heterodyne archi-

tecture won't be able to pickup the conducted emissions due to the spectrum analyzer's noise floor being higher than the noise floor from the Rhode&Schwarz EMI receiver.

The FIR Compiler on Zynq-7000 from Zedboard does not support coefficient arrays larger than 160, and the number of coefficients set in FIR Compiler affects the filter's performance and precision. As a result, the noise floor is higher than required. When attempting to use a filter with an attenuation of -80 dB after the cut-off band, the array of coefficients increased to 369, making optimization impossible. Figure 61 depicts the filter implementation problem.



(a) Desired filter characteristics



(b) Filter optimized implementation values

Figure 61: Filter optimized implementation

The length of the coefficients array is referred to as taps in Figure 1, and the frequencies, gains, and ripples are shown in detail. Because of the Zynq-7000 limitations, this array length is not supported, making it impossible to implement this filter on it.

Chapter 6: Conclusion

The main goal set out for the development of this thesis has been achieved in regards to building a custom, low frequency spectrum analyzer on top of a FPGA platform. The device is capable of capturing signals in frequency domain and display the power level in between 9 kHz and 108 MHz, intended for EMC measurements conducted at low frequencies. After the development of the spectrum analyzer and all the research made, one important conclusion is that this kind of measuring equipment is extremely important in these days, as devices are more susceptible to EMI. Power supplies are becoming more efficient at supplying massive amounts of power while maintaining high efficiency levels across the entire load range, at the expense of increased frequency and variable duty-cycle times. Although this is excellent in terms of efficiency, it is not in terms of EMI. That being said, current electronic devices must be designed with EMC concerns in mind in order to fulfill quality expectations and deliver the functionality required from the device. Following an examination of the many implementations of this measuring equipment, it was determined that the criteria for EMC testing is difficult to achieve since the signals are unknown, and have very low power, close to thermal noise floor. As a result, the lowest noise floor and maximum resolution bandwidth are necessary in order to view all of the components of an electromagnetic signal in detail, as this signal cannot be predicted. This project was ambitious given the criteria, nevertheless, it contains significant digital signal processing techniques that are highly relevant in today's technology, and using these techniques expanded the previously obtained knowledge.

The heterodyne design was the appropriate approach for increasing frequency resolution in the implemented system. This architecture allowed for the spectrum to be partitioned into several bandwidths, increasing the frequency resolution. Because the spectrum analyzer's criteria were quite demanding, this architecture provided a method of achieving them.

The most challenging step was being able to modify the reference design provided by Analog Devices in order to develop a custom spectrum analyzer that met the specifications stated in the introduction. The AD9467 IP Core has been designed to function better with Analog Devices DMA IP Core, and several changes to the system would be required in order to function with the Xilinx DMA.

The AXI communication protocol had to be adapted for all the IP Cores used in the spectrum computation as well as any extra calculations related to the heterodyne architecture. The IP Cores in the AD9467 Evaluation board reference design are configured to work with AXI4, and the IP Cores added to compute

the spectrum and implement the heterodyne architecture work exclusively with AXI4_Stream, thus the reference design was altered to work with AXI4_Stream to get both working.

To configure each IP Core, a data array containing values that would set specific configurations had to be set in the IP Core's configuration input, because the final system required a custom configuration. The DDS Compiler needed a data array, with the data at a certain position of the array telling the IP Core on which frequency the signal would be sent. Because there were several needed frequencies for the heterodyne architecture's local oscillator, it was essential to design a multiplexer to choose the appropriate frequency. The DDS Compiler signal was then mixed with the ADC signal in order to shift the spectrum to the right. Regarding the filter IP Core, FIR Compiler, it was used to filter one of the components from the output signal from the mixer, allowing a region of the spectrum to be selected. Because this IP Core is quite complete in terms of functionalities, but also rather complicated, it required some research into FPGA digital filter concepts and algorithms. The FFT IP Core saved a great deal of effort in terms of implementation, as the largest amount of FFT points is always desired for these types of applications. A FFT is relatively simple to implement when the characteristics of the signal to be analyzed or used are known, because in this case only a few FFT points are required, as opposed to a spectrum analyzer, which requires the maximum number of points possible to achieve precision and resolution. Following the FFT IP Core, the IP Core that implements the CORDIC algorithm was used. This IP Core was used to compute the signal's power at each frequency. Because the FFT will only be applied to the signal obtained from the heterodyne arquitecture, which is just a section of the signal, the frequency resolution is enhanced, as there are less frequency components to compute.

In terms of software, the reference design provided libraries that included all of the functions implemented for the AD9467, AD9517, and DMA. All of these functions were critical in the software component since they were employed in the AD9467 Evaluation Board's settings and calibrations. In the software component, the spectrum data calculated in the PL is instructed to be transferred to the PS through a DMA transfer. The information is then converted to real-world values. Regarding the AD9467 clock source, AD9517 would be a good choice, but it had fluctuations that would jeopardize the system's spectrum computations. The spectrum analyzer development was hindered by a malfunctioning voltage regulator on the evaluation board, which was responsible for supplying power to the SPI level translator. All SPI communications were compromised, making it impossible to configure the ADC and the AD9517 clock. The solution for this problem was to remove the voltage regulator from the board and use an external power source to directly power the SPI level shifter.

Owing to the limits of the provided hardware, it was not possible to meet the noise floor criteria. The

heterodyne design was not originally envisioned, but it was the best choice for completing the project. Instead of using DDS Compilers, a different ADC with two channels could be used so that one of them can be used to input the signal from an external local oscillator needed to mix with the signal meant to analyze. This has the potential to reduce the resources used on the FPGA, but even though this is a possible solution, it is not ideal as it would require the use of external equipment. Another option for lowering the noise floor is to implement and apply the filter on the input signal before sending it to the ADC, as the FIR Filter used increased the noise floor significantly due to the lack of BRAM, making it impossible to set the ideal arguments for its characteristic. Regarding the filter, it would be an improvement as well the implementation of other types of windows besides the rectangular, as each window has different characteristics that may be used according to the use case intended. Implementing an FFT in verilog for the purpose of having a large spectrum bandwidth and a high resolution bandwidth is very laborious and time consuming, as it would require a large number of FFT points, so using an IP Core capable of computing this number of FFT points is less time consuming and easier. The maximum points for the Intel FFT IP Core are 262144, which is four times the maximum points for the Vivado FFT IP Core. In this instance, an Intel FPGA development board should be considered for this application. Another method of improving the system is to use timing strategies during the hardware accelerator's development, as timing problems might jeopardize the system's functionality, and this project encountered timing faults after implementing the heterodyne architecture. In the end, the developed spectrum analyzer was capable of capturing eletromagnetic signals, and calculate its spectrum between 9kHz and 108MHz. This project can be used as a proof-of-concept for a future project, owing to its utility in EMC debugging and its low-cost approach.

References

[1] I. Tektronix, "Fundamentals of real-time spectrum analysis." https://download.tek.com/document/37W_17249_5_HR_Letter.pdf, 2013.

[2] I. Keysight Technologies, "Different types of analyzers." https://rfmw.em.keysight.com/wireless/helpfiles/89600b/webhelp/subsystems/concepts/Content/concepts_types_spec_an.htm, 2015.

[3] S. Dawwd and S. Nori, "Reduced area and low power implementation of fft/ifft processor," *Iraqi Journal for Electrical and Electronic Engineering*, vol. 14, pp. 108–119, 12 2018.

[4] A. S. Garth D'Abreu, Craig Fanning, "Automotive emc testing: Cispr 25, iso 11452-2 and equivalent standards." https://incompliancemag.com/article/automotive-emc-testing-cispr-25-iso-11452-2-and-equivalent-standards, 01 2016.

[5] Avnet, "Zedboard (zynq evaluation and development) hardware user's guide." https://files.digilent.com/resources/programmable-logic/zedboard/ZedBoard_HW_UG_v2_2.pdf, 2014.

[6] A. Devices, "Ad9467 native fmc card / xilinx reference design." https://wiki.analog.com/resources/fpga/xilinx/fmc/ad9467, 2021.

[7] Xilinx, "Cordic v6.0." https://www.xilinx.com/support/documentation/ip_documentation/cordic/v6_0/pg105-cordic.pdf, 2021.

[8] C. Toole, "Intro to AXI Protocol: Understanding the AXI interface - SoC Design blog - System - Arm Community," 2016.

[9] Xilinx, "Fast fourier transform v9.0." https://www.xilinx.com/support/documentation/ip_documentation/xfft/v9_0/pg109-xfft.pdf, 2017.

[10] Z. Zou, Y. Liu, F. Zhao, and M. Peng, "Design and data processing of a real-time power quality monitoring instrument," in *2009 9th International Conference on Electronic Measurement Instruments*, pp. 4–353–4–357, 2009.

[11] E. Flaxer, "Smart filament emission controller for supersonic gas chromatography-mass spectrometry using real time digital signal processor," in *2012 5th European DSP Education and Research Conference (EDERC)*, pp. 105–108, 2012.

[12] R. Amirtharajah, J. Collier, J. Siebert, B. Zhou, and A. Chandrakasan, "Dsps for energy harvesting sensors: Applications and architectures," *IEEE Pervasive Computing*, vol. 4, pp. 72–79, 07 2005.

[13] A. Combo, A. Silva, P. Varela, M. Manso, J. Sousa, and C. Varandas, "Upgrade of the data acquisition and control system of the asdex upgrade microwave reflectometer," *Nuclear Science, IEEE Transactions on*, vol. 53, pp. 918 – 922, 07 2006.

[14] A. Rodrigues, C. Correia, C. Varandas, and F. Schneider, "Multiple dsp system for real time parallel processing and feedback control on fusion experiments," *Fusion Engineering and Design*, vol. 43, no. 3, pp. 401–405, 1999.

[15] S. Aiello, A. Anzalone, M. Bartolucci, G. Cardella, S. Cavallaro, E. DeFilippo, S. Feminò, M. Geraci, F. Giustolisi, P. Guazzoni, M. Iacono Manno, G. Lanzalone, G. Lanzanò, S. Lo Nigro, G. Manfredi, A. Pagano, M. Papa, S. Pirrone, G. Politi, F. Porto, F. Rizzo, S. Sambataro, L. Sperduto, C. Sutera, and L. Zetta, "Data acquisition and real-time computing by a dsp-based system," *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, vol. 136-138, pp. 1172–1176, 1998. Ion Beam Analysis.

[16] P. Pinto, J. Gouveia, and P. Ramos, "Development, implementation and characterization of a dsp based data acquisition system with on-board processing," *Acta IMEKO*, vol. 4, pp. 19–25, 02 2015.

[17] N. M. Önder, M. C. Çakmak, and Ünver, "Fpga based data acquisition and test system design for diagnostic testing," in *2017 IEEE AUTOTESTCON*, pp. 1–5, 2017.

[18] K. C. Rovers, "Front-end research for a low-cost spectrum analyser : receiver system and topology study," June 2006.

[19] N. S. H. Sibu Thomas, "A study on basics of a spectrum analyzer," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, June 2013.

[20] I. Technology, "2016 real time spectrum analyzer guide," 2016.

[21] M. Heideman, D. Johnson, and C. Burrus, "Gauss and the history of the fast fourier transform," *Archive for History of Exact Sciences*, vol. 34, pp. 265–277, 01 1985.

[22] A. Gasmi, "What is fast fourier transform?," 08 2022.

[23] G. Proakis, "Digital signal processing," 1996.

[24] J. S. Walther, "The story of unified cordic," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 14, 06 2000.

[25] P. Choudhary and A. Karmakar, "Cordic based implementation of fast fourier transform," in *2011 2nd International Conference on Computer and Communication Technology (ICCCT-2011)*, pp. 550–555, 2011.

[26] K. McMillin, "What are emc and emi?." `https://www.digikey.pt/en/blog/what-are-emc-and-emi`, 10 2021.

[27] Xilinx, "Dds compiler v6.0." `https://www.xilinx.com/support/documentation/ip_documentation/dds_compiler/v6_0/pg141-dds-compiler.pdf`, 2021.

[28] Xilinx, "Fir compiler v7.2." `https://www.xilinx.com/support/documentation/ip_documentation/fir_compiler/v7_2/pg149-fir-compiler.pdf`, 2021.

[29] CieNTi, "Serial port plotter." `https://github.com/CieNTi/serial_port_plotter`, 2018.

[30] P. Isza, "Tfilter." `http://t-filter.engineerjs.com`, 2011.

[31] Brüel and Kjær, "Brüel and kjær technical review." `https://www.bksv.com/media/doc/Bv0031.pdf`, 1987.

[32] M. H. Weik, *Parseval's theorem*, pp. 1231–1231. Boston, MA: Springer US, 2001.

Appendix A: AD9467 Board malfunction

The first results regarding the signal amplitude were not the expected. Even though the bin number matched the signal's frequency, its amplitude was not correct. The system had an odd behavior, because if the amplitude of the input signal was increased, the output value from the system would decrease, and vice-versa. According to Parceval's theorem [32], the power of the signal in the time domain equals the power of the transformed signal in the frequency domain, therefore the amplitude should be equal to the Peak-to-Peak Voltage (Vpp).

To troubleshoot this behavior, Chipscope was used to analyze data from the system while the software was running. The data from the AD9467 IP Core was incorrect, which caused the FFT IP Core to receive incorrect data. The signal peaks were inverted, and its value increased when the AD9467 input signal decreased, and vice versa, as described previously.

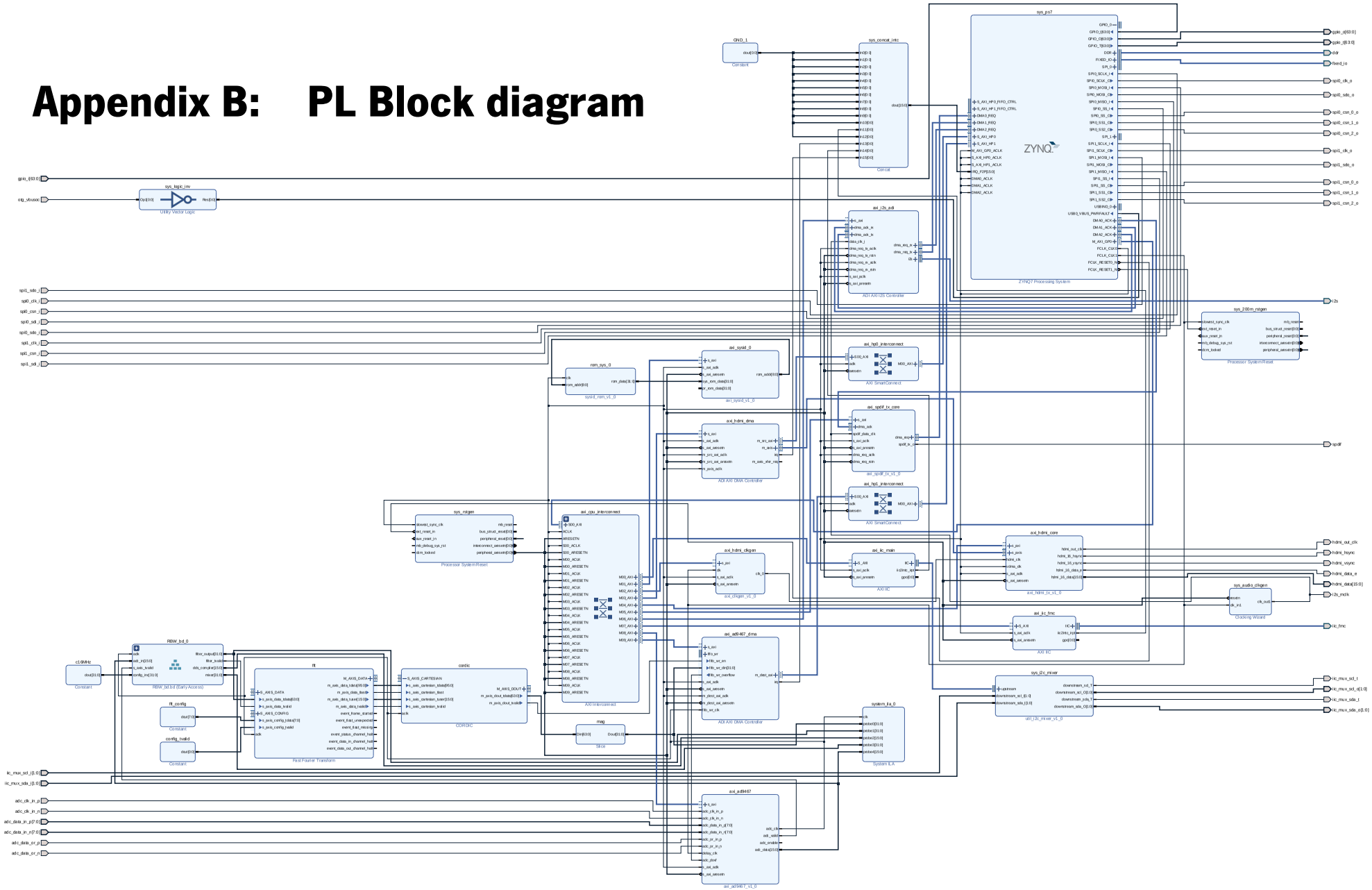
Following some tests, it was determined that the root of the problem was the SPI communication, due to the inability to change configurations on the AD9467 via SPI. The AD9467 output is set in offset binary format by default, and it had to be set to two's complement format because the FFT IP Core datasheet states that its input does not support offset binary.

To troubleshoot this, with the aid of an oscilloscope, the SPI signals were measured from the Zedboard until the AD9467 in order to get to the core of the problem. The SPI signal was operational until it reached *ADG3304BCPZ*, a bi-directional logic level translator on the AD9467 board, used to provide an interface between components that operate at different voltage levels. At the level translator's outputs, there weren't any signals. Further investigation revealed the problem was with the voltage level of the outputs (ADC side), where a voltage of 3.3V was expected, the values read were of 1.3V, hence the failure in communication. This component needs 3.3V on pin 18, and when measured, it had approximately 1.3V.

In order to understand why it wasn't being powered correctly, further measurements were made on the AD9467 board. By measuring the voltage regulator responsible for powering the level shifter, it was concluded that it wasn't operating correctly, affecting the SPI communication, as the ADC is expecting a 3.3V referenced bus. The solution was to remove the voltage regulator and feed the level translator 3.3V using an external power source.

After these modifications, the tests to the system while running the software component were resumed.

Appendix B: PL Block diagram



Appendix C: Heterodyne architecture module block diagram