A Massive MIMO Digital Beamforming mm-Wave Transceiver for 5G

A Graduation Project Thesis

Submitted to

Electronics & Communications Engineering Department

Faculty of Engineering

Ain Shams University

Submitted by

Asmaa Anwar Hamed Eissa
Donia Mohamed Adel Elsaid
Jumana Ahmed Mohamed
Moamen Khaled Moselhi Mohamed
Mohamed Abdallah Ali Mohamed
Mohamed Ahmed Fares Zyan
Mohamed Hossam Rafik
Shady Emad Zaki
Yousif Ahmed Ali Mohamed
Ziad Ahmed Eped Mohamed

Under The Supervision of

Dr. Micheal Ibrahim

Dr. Theodora Mamdouh

January 2025

Contribution Table

Literature review	All members
MATLAB Simulations	All members
System modelling of Interpolation and Decimation blocks	Asmaa Anwar Hamed Eissa Donia Mohamed Adel Elsaid Shady Emad Zaki Mohamed Hossam Rafik
System modelling of the Digital Delta Sigma modulator	Jumana Ahmed Mohamed Mohamed Ahmed Fares Zyan Yousif Ahmed Ali Mohamed
System modelling of the CWM, DUC, DDC and interleaver blocks	Moamen Khaled Moselhi Mohamed Mohamed Abdallah Ali Mohamed Ziad Ahmed Eped Mohamed
Digital implementation of Tx	Asmaa Anwar Hamed Eissa Donia Mohamed Adel Elsaid Jumana Ahmed Mohamed Mohamed Ahmed Fares Zyan Shady Emad Zaki
Digital implementation of Rx	Moamen Khaled Moselhi Mohamed Mohamed Abdallah Ali Mohamed Mohamed Hossam Rafik Yousif Ahmed Ali Mohamed Ziad Ahmed Eped Mohamed

Table of Contents

List of Figures 6
List of Tables 11
List of Abbreviations 12
Abstract 14

Chapter 1: Introduction 15
1.1 Motivation 15
1.2 Technological advancements 16
1.3 Document Breakdown 18

Chapter 2: Literature Review 19
2.1 MIMO 19
2.1.1 What is a MIMO system 19
2.1.2 The transition from SISO to MIMO 19
2.1.3 Why MIMO 21
2.1.4 Single-User MIMO 22
2.1.5 Multi-User MIMO 22
2.1.6 What is Massive MIMO 24
2.1.7 Development of MIMO Technology 26

2.2 Beamforming 27
2.2.1 What is Beamforming 27
2.2.2 Beamforming in MIMO systems 27
2.2.3 Benefits of beamforming 28
2.2.4 Beamforming techniques and types 29

2.3 Digital Beamforming 32
2.3.1 Phased Arrays 32
2.3.2 Steering vector 36
2.3.3 Delay and Sum Beamformer 38
2.3.4 MVDR/Capon Beamformer 40
2.3.5 LCMV Beamformer 43
2.3.6 Subspace Techniques 44
2.3.6.1 MUSIC 44
2.3.6.2 ESPRIT 45

Chapter 3: MATLAB Simulations 48

3.1 Delay and sum 48

3.2 MVDR 50

3.3 Delay and sum vs MVDR comparison 51

3.4 LCMV 52

3.5 MUSIC 53

3.6 ESPRIT 55

Chapter 4: System Modeling 57

4 .1 System overview 57

4.2 Receiver Model 60

4.2.1 Sigma Delta ADC 61

4.2.2 Interleaver 62

4.2.3 Digital Down Converter 63

4.2.4 CWM 65

4.2.5 Decimation filter 67

4.3 Transmitter Model 89

4.3.1 CWM 90

4.3.2 Interpolation filter 94

4.3.3 Digital Delta Sigma Modulator 100

4.3.4 Digital Up Converter 106

4.4 System integration & testing environment 107

Chapter 5: Digital Design 110

5.1 RTL Implementation 111

5.1.1 Clock divider 111

5.1.2 RTL implementation for Rx blocks 112

5.1.3 RTL implementation for Tx blocks 121

5.2 RTL verification 128

5.3 Logical Synthesis 129

5.4 Formal Verification (Post-Synthesis) 131

5.5 Design for Testability (DFT) 132

5.6 Formal Verification (Post-DFT) 135

5.7 Place and Route (PNR) 136

Conclusion 146
References 147

List of Figures

Figure 1 - Number of connected devices and data traffic from 2016 to 2022
Figure 2 – MIMO antenna system
Figure 3 – SISO implementation
Figure 4 – SIMO implementation
Figure 5 – MISO implementation
Figure 6 – MIMO implementation
Figure 7 – MIMO as a receiver
Figure 8 – MIMO as a transmitter
Figure 9 – SU-MIMO
Figure 10 – MU-MIMO
Figure 11 – MU-MIMO uplink channel
Figure 12 – MU-MIMO downlink channel
Figure 13 – Massive MIMO downlink channel
Figure 14 – Antenna configuration for the LTE technologies
Figure 15 – Beamforming Example
Figure 16 – Beamforming antenna array Vs Sectored antenna array
Figure 17 – Beamforming array
Figure 18 – Beamforming Techniques
Figure 19 – Analog Beamforming Receiver
Figure 20 – Digital Beamforming Receiver
Figure 21 – Hybrid Beamforming Receiver
Figure 22 – Antenna array model
Figure 23 – Single isotropic antenna radiation pattern
Figure 24 – Two isotropic antennas radiation pattern
Figure 25 – Two isotropic antennas radiation pattern (zoomed out)
Figure 26 – Two isotropic antennas with spacing λ radiation pattern
Figure 27 – Three isotropic antennas radiation pattern
Figure 28 – Two isotropic antennas with phase shift radiation pattern
Figure 29 – Eight isotropic antennas with beam steering
Figure 30 – ULA
Figure 31 – Delay calculation
Figure 32 – Delay and sum beamforming
Figure 33 – Decomposing the network into two sub-networks
Figure 34 – Transmitted signal
Figure 35 – Received signals at the ULA
Figure 36 – Delay and sum beamformed signal
Figure 37 – Delay and sum DOA metric rectangular plot
Figure 38 – Delay and sum DOA metric polar plot
Figure 39 – MATLAB DOA calculation
Figure 40 – Transmitted signal
Figure 41 – Received signals at the ULA

Figure 42 – MVDR beamformed signal
Figure 43 – MVDR DOA metric rectangular plot
Figure 44 – MVDR DOA metric polar plot
Figure 45 – MATLAB DOA calculation
Figure 46 – Delay and sum Vs MVDR DOA metric rectangular plot
Figure 47 – Delay and sum Vs MVDR DOA metric polar plot
Figure 48 – LVMV DOA metric rectangular plot
Figure 49 – LCMV DOA metric polar plot
Figure 50 – MUSIC DOA metric rectangular plot
Figure 51 – MUSIC DOA metric polar plot
Figure 52 – MUSIC number of input signals estimation
Figure 53 – Underestimating the number of input signals
Figure 54 – Overestimating the number of input signals
Figure 55 – MUSIC for closely separated signals
Figure 56 – ESPRIT DOAs MATLAB calculation
Figure 57 – MUSIC & ESPRIT DOA metric
Figure 58 – (a) IF-sampling DBF and (b) its MUX-based implementation
Figure 59 – (a) DSP after decimation (b) BSP
Figure 60 – Bit-stream multiplication with a 2:1 MUX
Figure 61 – (a) DSP with multiple decimators (b) BSP with a single decimator
Figure 62 – Proposed RX model
Figure 63 – ADC output in time domain
Figure 64 – DDC before adding IL
Figure 65 – DDC after adding IL
Figure 66 – IL block diagram initially
Figure 67 – Final IL block diagram
Figure 68 – Interleaver Simulink model
Figure 69 – DDC
Figure 70 – Three-level I/Q LO sequences
Figure 71 – DDC model
Figure 72 – DDC Simulink model
Figure 73 – Phase shifting with CWM
Figure 74 – CWM Simulink model
Figure 75 - The Decimation process
Figure 76 - The proposed Decimation Filter
Figure 77 – Gain Response of the single comb filter for N = 10
Figure 78 – Gain Response of the single comb filter for k = 1, 2, 3 and 4
Figure 79 – Pass band response of the single comb filter for k = 1, 2, 3 and 4
Figure 80 – Recursive algorithm for the comb filters
Figure 81 – Sampling rate and word length of the IIR filter in the recursive algorithm
Figure 82 – Non-Recursive algorithm for the comb filters
Figure 83 – Sampling rate and word length of the stage i in the non-recursive algorithm ($i = 1, 2, \dots M$)
Figure 84 – The magnitude response of a single stage comb filter

Figure 85 – The magnitude response of a 4-stage comb filter
Figure 86 – Simulink model of the comb filter
Figure 87 – Frequency response of the designed comb filter
Figure 88 – 4th band filter with $k = 10$
Figure 89 – Half band filter with $k = 5$ and $N = 11$
Figure 90 – Impulse response of the designed half band filter
Figure 91 – Frequency response of the designed half band filter
Figure 92 – Simulink model of the designed half band filter
Figure 93 – Frequency response of the designed half band filter
Figure 94 – Five-tap FIR filter
Figure 95 – SFG of a five-tab FIR filter
Figure 96 – SFG of a transposed five-tab FIR filter
Figure 97 – Rearranged SFG of a transposed five-tab FIR filter
Figure 98 – The block diagram of the transposed-form FIR filter
Figure 99 – The pipelined transposed-form FIR filter
Figure 100 – Simulink model of the transposed implementation of the comb filter
Figure 101 – Frequency response of the transposed implementation of the comb filter
Figure 102 – Simulink model of the transposed implementation of the half band filter
Figure 103 – Frequency response of the transposed implementation of the half band filter
Figure 104 – Polyphase decomposition with M = 2, 3 and 4
Figure 105 – Coefficients of the polyphase decomposition with M = 2, 3 and 4
Figure 106 – Decimation equivalent structure in multi rate systems
Figure 107 – Polyphase decomposition of a decimator
Figure 108 – Simulink model of the polyphase decomposition of the comb filter used in decimation
Figure 109 – Simulink model of the first 2 polyphase components of the comb filter used in decimation
Figure 110 – Simulink model of the polyphase decomposition of the half band filter used in decimation
Figure 111 – Simulink model of the polyphase components of the half band filter used in decimation
Figure 112 – Frequency response of the decimation filter
Figure 113 – Simulink model of the decimation filter
Figure 114 – Frequency Response of the Simulink model of the decimation filter
Figure 115 – simulation test in Simulink
Figure 116 – Input and Output signals
Figure 117 – Proposed Transmitter model
Figure 118 – CWM block diagram (Tx)
Figure 119 – CORDIC Algorithm Rotation
Figure 120 – Simulink model of CWM
Figure 121 – CORDIC in CWM
Figure 122 – Frequency domain of an up sampled signal
Figure 123 – The Interpolation process (L=5)
Figure 124- The proposed Interpolation Filter
Figure 125 – Interpolation equivalent structure in multi rate systems
Figure 126 – Polyphase decomposition of an interpolator
Figure 127 – Simulink model of the polyphase decomposition of the comb filter used in interpolation

Figure 128 – Simulink model of the first 2 polyphase components of the comb filter used in interpolation
Figure 129 – Simulink model of the polyphase decomposition of the half band filter used in interpolation
Figure 130 – Simulink model of the polyphase components of the half band filter used in interpolation
Figure 131 – Frequency response of the interpolation filter
Figure 132 – Simulink model of the interpolation filter
Figure 133 – Frequency Response of the Simulink model of the interpolation filter
Figure 134 – simulation test in Simulink
Figure 135 – Input and Output signals
Figure 136 - Block diagram of the DDSM with n-bit input x[n] and m-bit output y[n]
Figure 137 - Spectra in a DDSM with a sinusoidal input.
Figure 138 - A first-order EFB [2] DSM
Figure 139 - An n^{th} order EFB DS
Figure 140 – A third-order EFB DSM with four adder critical path
Figure 141 – A MASH 1-1 DSM
Figure 142 – A pipelined MASH 1-1 DSM with only one adder critical path
Figure 143 – Simulink model of the designed MASH 1-1-1 DSM
Figure 144 – Frequency response and SNR of the 1-1-1 MASH
Figure 145 – DUC Architecture
Figure 146 – DUC block diagram
Figure 147 – Simulink model of the DUC
Figure 148 – Testing environment for the system
Figure 149 – Transceiver chain
Figure 150 – BER calculation 1
Figure 151 – BER calculation 2
Figure 152 – BER calculation 3
Figure 153 – Data converter in Simulink
Figure 154 – Fixed point model in Simulink
Figure 155 – Clock divider elaborated design
Figure 156 – Interleaver Simulink model
Figure 157 – Interleaver elaborated design
Figure 158 – DDC Simulink model
Figure 159 – DDC elaborated design
Figure 160 – RX CWM Simulink model
Figure 161 – RX CWM elaborated design
Figure 162 – Comb filter in the decimation
Figure 163 – E_0 in the comb filter
Figure 164 – Comb filter E_0 elaborated design
Figure 165 – Comb filter elaborated design
Figure 166 – Down sampler elaborated design
Figure 167 – Half band filter in the decimation
Figure 168 – E_1 in the half band filter
Figure 169 – Half band E_1 elaborated design
Figure 170 – Decimation half band filter elaborated design

Figure 171 – decimation filter elaborated design
Figure 172 – Rx chain elaborated design
Figure 173 – 4 Rx chains Elaborated design
Figure 174 – MIMO Rx elaborated design
Figure 175 – CWM Simulink model
Figure 176 – Weight multiplication elaborated design
Figure 177 – CORDIC elaborated design
Figure 178 – Tx CWM elaborated design
Figure 179 – Interpolation filter Simulink model
Figure 180 – Comb filter in interpolation
Figure 181 – Half band filter in interpolation
Figure 182 – Interpolation comb filter elaborated design
Figure 183 – Interpolation half band filter elaborated design
Figure 184 – Interpolation filter elaborated design
Figure 185 – A single stage of the MASH DDSM Architecture
Figure 186 – Single stage MASH DDSM Architecture elaborated design
Figure 187 – Pipelined MASH Noise Cancellation network
Figure 188 – Pipelined MASH Noise Cancellation network elaborated design
Figure 189 – DDSM elaborated design
Figure 190 – DUC Simulink model
Figure 191 – DUC elaborated design
Figure 192 – Transmitter chain elaborated design
Figure 193 – Simulink In/Out extraction
Figure 194 – decimation verification
Figure 195 – number of wrong outputs
Figure 196 – PVT variations [196]
Figure 197 – Tx LEC compare results
Figure 198 – Rx LEC compare results
Figure 199 – Scan Flip Flop [41]
Figure 200 – DFT violations report
Figure 201 – Tx coverage report
Figure 202 – Rx coverage report
Figure 203 – Tx LEC compare results
Figure 204 – Rx LEC compare results
Figure 205 – Pre placement activities in PNR [45]
Figure 206 – Tx Power planning
Figure 207 – Rx Power planning
Figure 208 – Tx Placement
Figure 209 – Rx Placement
Figure 210 – Tx Pre CTS Timing analysis
Figure 211 – Rx Pre CTS Timing analysis
Figure 212 – Tx CTS
Figure 213 – Rx CTS

Figure 214 – Tx Post CTS timing analysis & ECO optimization
Figure 215 – Rx Post CTS timing analysis & ECO optimization
Figure 216 – DRC violations
Figure 217 – Tx Post routing timing analysis & ECO optimization
Figure 218 – Rx Post routing timing analysis & ECO optimization
Figure 219 – Tx Chip Layout after Filler Cell Insertion
Figure 220 – Rx Chip Layout after Filler Cell Insertion

List of Tables

Table 1 – Synthesis constraints
Table 2 – Tx synthesis results
Table 3 – Rx synthesis results
Table 4 – Tx DFT results
Table 5 – Rx DFT results
Table 6 – Rx PNR results
Table 7 – Rx PNR results

List of Abbreviations

mm-Wave	Millimeter wave
MIMO	Multiple input multiple output
SISO	Single input single output
SIMO	Single input multiple output
MISO	Multiple output single input
SU-MIMO	Single-user Multiple input multiple output
MU-MIMO	Multi-user Multiple input multiple output
OFDM	Orthogonal Frequency Division Multiplexing
RF	Radio Frequency
ULA	Uniform linear array
AOA	Angle of arrival
DOA	Direction of arrival
SOI	Signal of interest
Tx	Transmitter
Rx	Receiver
DPF	Digital Beamforming
CTBPDSM	Continuous Time Band Pass Delta Sigma Modulator
BSP	Bit Stream Processing
DDC	Digital Down Converter
MUX	Multiplexer
IF	Intermediate Frequency
ADC	Analog to Digital Converter
DAC	Digital to Analog Converter
SNR	Signal to Noise Ratio
I/Q	In-phase/Quadrature
FR2	Frequency Range 2
3GPP	3rd Generation Partnership Project
OSR	Over Sampling Ratio
RTL	Register Transfer Level
DSP	Digital Signal Processing
IL	Interleaver
LO	Local Oscillator
LUT	Look-up table
CWM	Complex Weight Multiplication
MA	Moving Average
VLSI	Very-Large-Scale Integration
FIR	Finite Impulse Response
IIR	Infinite Impulse Response
DUC	Digital-Up-Converter
DDSM	Digital Delta Sigma Modulator
CORDIC	Coordinate Rotational Digital Computer
EFB	Embedded Feedback
MASH	Multi Stage Noise Shaping
BER	Bit Error Rate

ATPG	Automatic Test Pattern Generation
CORDIC	Coordinate Rotation Digital Computer
CTS	Clock Tree Synthesis
DFT	Design for Testability
DRC	Design Rule Check
ECO	Engineering Change Order
FPGA	(Not explicitly found but commonly related)
HDL	Hardware Description Language
I/O	Input/Output
LO	Local Oscillator
LEC	Logical Equivalence Checking
PNR	Place and Route
PVT	Process, Voltage, Temperature
RTL	Register Transfer Level
SD	Standard Delay (SDF File)
SDC	Synopsys Design Constraints
SDF	Standard Delay Format
SPEF	Standard Parasitic Exchange Format
SQNR	Signal-to-Quantization-Noise Ratio
STA	Static Timing Analysis

Abstract

There are **three timeless truths** in the field of wireless communications:

- Demand for wireless throughput, both mobile and fixed, **will always increase**.
- The available electromagnetic spectrum **will never increase**, and the most desirable frequency bands that can propagate into buildings and around obstacles and that are unaffected by weather **constitute only a small fraction of the entire spectrum**.
- Communication theorists and engineers will always be pressured to invent or to discover breakthrough technologies that **provide higher spectral efficiency**. [1]

To address these challenges we will investigate both **MIMO** systems and **Beamforming techniques** in **5G mm-wave**.

MIMO is considered a breakthrough technology in the area of wireless communication because it achieves an increase in data rates, channel capacity and spectral efficiency by constructing base stations with large number of antennas that simultaneously communicate with multiple spatially separated user terminals over the same frequency resource and exploit multipath propagation.

Beamforming is a signal processing technique to steer, shape, and focus signals using an array of antennas toward a desired direction. It's used to increase SNR of the desired signals, null out interferers, shape beam patterns, or even transmit/receive multiple data streams at the same time and frequency.

5G mm-Wave (millimeter-wave) refers to the higher range of radio frequencies supported by 5G (beyond 4G frequencies). This is termed 5G FR2 (Frequency Range 2) and extends above 24 GHz. The lower 5G FR1 bands are from 410 MHz to 7125 MHz. 5G mm-Wave supports high capacity and fast data throughput with low latency.

Ch 1: Introduction

1.1 Motivation

Wireless communication technology has fundamentally changed the way we communicate. The time when telephones, computers, and Internet connections were bound to be wired, and only used at predefined locations, has passed. These communications services are nowadays wirelessly accessible almost everywhere on Earth, thanks to the deployment of cellular wide area networks (e.g., based on the GSM1, UMTS2, and LTE3 standards), local area networks (based on different versions of the Wi-Fi standard IEEE 802.11), and satellite services.

Wireless connectivity has become an essential part of the society—as vital as electricity—and as such the technology itself spurs new applications and services. We have already witnessed the streaming media revolution, where music and video are delivered on demand over the Internet. The first steps towards a fully networked society with augmented reality applications, connected homes and cars, and machine-to-machine communications have also been taken. Looking 15 years into the future, we will find new innovative wireless services that we cannot predict today.

The motivation for this project arises from the critical need to address the limitations and challenges faced by current wireless communication systems.

As the demand for higher data rates, increased capacity and improved user experience continues to grow existing technologies are increasingly struggling to meet these requirements.

Several key factors drive the motivation for integrating **Massive MIMO**, **Beamforming**, and **mm-Wave** technologies into modern wireless networks.

Increased demand for higher data rates and capacity

One of the main drivers of wireless communication system growth is the demand for higher data rates and capacity from users and applications. With the proliferation of smartphones, tablets, laptops, and other devices, the wireless traffic is expected to grow exponentially in the coming years. Moreover, new applications such as video streaming, online gaming, cloud computing, and virtual reality require higher bandwidth and lower latency than traditional voice and text services.

Industry projections indicate that global mobile data traffic will continue to grow at a rapid pace. Meeting these future demands necessitates advancements in wireless technology that can offer significantly higher capacity and throughput.

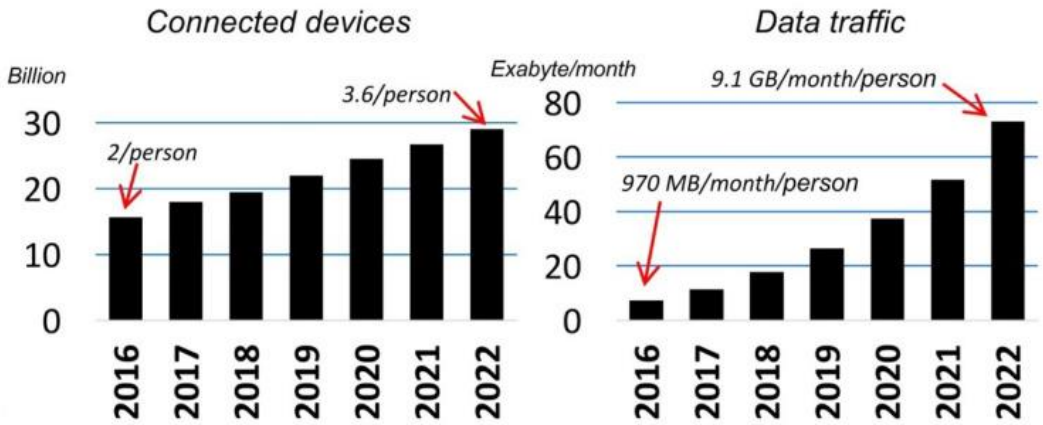

Figure 1 - number of connected devices and data traffic from 2016 to 2022 [9]

To meet these requirements, wireless communication systems need to adopt advanced technologies such as multiple-input multiple-output (MIMO), orthogonal frequency-division multiplexing (OFDM), and beamforming, as well as explore new frequency bands such as millimeter wave (mm-Wave) and terahertz (THz).

Increased demand for more spectral efficiency

• Limited Spectrum Availability:

The radio frequency spectrum is a finite resource, and the lower frequency bands (sub-6 GHz) commonly used for wireless communications are becoming increasingly congested. This congestion limits the available bandwidth for new services and applications.

• Higher Frequencies:

The underutilized mm-Wave spectrum (30-300 GHz) offers a vast amount of bandwidth, which can be leveraged to alleviate spectrum scarcity and provide high data rates. However, mm-Wave signals have different propagation characteristics and require new approaches to be effectively utilized.

1.2 Technological advancements

MIMO

Multiple-input-multiple-output communication technology is a breakthrough in wireless communication that has provided greater scalability and the ability to serve more users with lower latency services.

In its infancy, MIMO was an innovative technology that employed multiple antennas at both transmitting and receiving ends to increase the data transfer capacity. However, with the addition of multiplexing techniques like OFDM, antenna arrays using MIMO techniques are

used for simultaneously transmitting and receiving multiple data streams over the same radio channel, which is aided by multipath propagation.

Spatial Multiplexing

Spatial multiplexing involves sending multiple spatial streams through multiple antennas, and these streams are separated at the receiver through spatial processing. Since the receiver decodes the transmitted streams individually, data throughput can be increased for a fixed channel bandwidth. MIMO spatial multiplexing has the ability to boost spectral efficiency without significantly diminishing the link's robustness. However, diversity gain is lost in this technique.

Beamforming

The use of beamforming in MIMO communications involves focusing a signal in a particular direction so that the greatest possible gain is achieved at the receiving end. There are three beamforming techniques:

- Analog, which would be performed with a phased array
- Digital, which uses precoding with modulated data streams to construct a beam pattern
- Hybrid, where analog and digital are combined and multiplexed spatially/temporally

Different beamforming methods used with spatial multiplexing/diversity will require different signal processing methodologies to precode and decode signals. This would need to be implemented with a specialty chipset or an FPGA.

Single-User (SU) and Multi-User MIMO

SU-MIMO supports a single device at a time by using radio communication layers from multiple antennas. To achieve the highest possible user spectral efficiency, SU-MIMO transmissions utilize time-frequency resources that are exclusively allocated to a single user device. MU-MIMO, on the other hand, allocates multiple users to a single time-frequency resource. This is a significant advantage as compared to SU-MIMO, particularly in channels with spatial correlation.

OFDM in MIMO Communication

OFDM is a form of multicarrier modulation in which data is sent at a reduced rate using parallel subcarriers. The data is first split and superimposed onto numerous frequency carriers prior to transmission, and then it is merged at the receiving end. Uniting MIMO and OFDM leads to an increase in spectral efficiency as spatial multiplexing gain offered by MIMO is used in combination with multi-carrier modulation. The diversity gain achieved by MIMO technology can also increase connection stability and provide a high quality of service (QoS). The practical

implementation of OFDM MIMO is no longer as complex as it could have been thanks to advances in baseband digital signal processing and VLSI technology.

Reconfigurable antennas in MIMO

Reconfigurability adds flexibility to a MIMO communication system and significantly improves transmission throughput. Polarization, frequency, and radiation patterns are fixed in the case of conventional MIMO antennas. In contrast, reconfigurable antennas can serve as a booster for MIMO systems because their radiation properties (polarization, radiation pattern, frequency) can be reconfigured in response to environmental and system changes. [14]

1.3 Document Breakdown

This document is divided into five chapters:

Chapter one, the introduction, discusses the motivation behind this project, emphasizing the increased demand for higher communication speeds and the technological advancements that enable the use of Massive MIMO and beamforming in modern communication systems.

Chapter two is a literature review that explores MIMO systems, their functionality, and the development of MIMO technology. It provides a general explanation of beamforming, highlighting its benefits, various types, and approaches. It then focuses on digital beamforming, starting with phased arrays and steering vector explanations before detailing how different digital beamforming techniques, such as delay-and-sum, MVDR, LCMV, MUSIC, and ESPRIT, operate.

Chapter Three presents the MATLAB-based simulations of the proposed digital beamforming techniques. It details the simulation parameters, mathematical models, and key equations, accompanied by the resulting performance metrics and illustrative figures.

Chapter Four focuses on the system-level modeling of the proposed receiver (Rx) and transmitter (Tx) architectures. It provides a comprehensive overview of the system blocks, hardware simulation results, and the finalized design outcomes.

Chapter Five discusses the digital hardware implementation of the system. This includes the Register Transfer Level (RTL) design, logic synthesis, Design for Testability (DFT), formal equivalence checking using Formality, and the Place and Route (PNR) process.

Ch 2: Literature Review

2.1 MIMO

2.1.1 What is a MIMO system?

MIMO stands for Multiple-Input Multiple-Output. A MIMO antenna is an antenna system that uses multiple antennas to transmit and receive data simultaneously, thereby increasing the capacity and performance of wireless communication systems.

In a MIMO antenna system, the transmitter and receiver are equipped with multiple antennas. These antennas are strategically placed to create multiple spatial channels for data transmission. Each antenna in the transmitter sends a different copy of the data simultaneously, and each antenna in the receiver receives the copies of the data. The receiver then combines these received signals to improve the signal quality and increase the data throughput. [6]

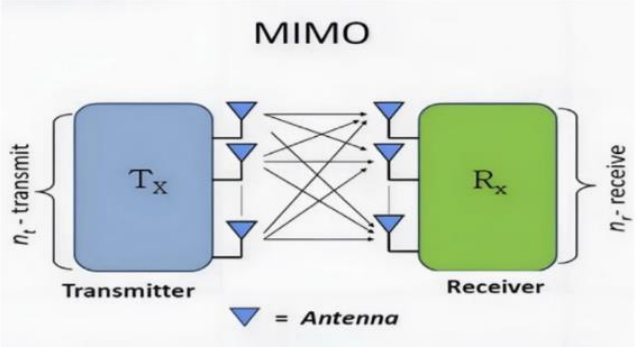

Figure 2 – MIMO antenna system [6]

2.1.2 The transition from SISO to MIMO

SISO provides a unique path between the Tx and Rx antennas and transmits one signal. In a wireless system, each signal is transmitted over one spatial stream. Apparently, such transmission is **unreliable** and **rate limited.**

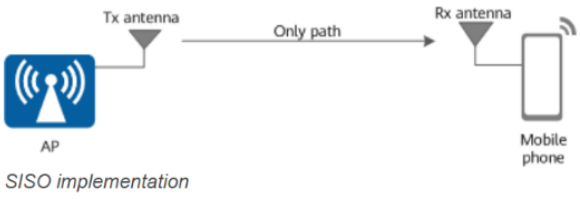

Figure 3 – SISO implementation [7]

To achieve higher **reliability**, **SIMO** and **MISO** are introduced:

Single-input multiple-output (SIMO). That is, one antenna is added on the receiver, as shown Fig. 4, so that two signals can be received concurrently. Two signals containing the same data are sent from one Tx antenna. If one signal is partially lost, the receiver may still obtain complete data from the other signal. Although the capacity remains unchanged (still one path in use), the reliability is doubled. This mode is also known as **receive diversity**.

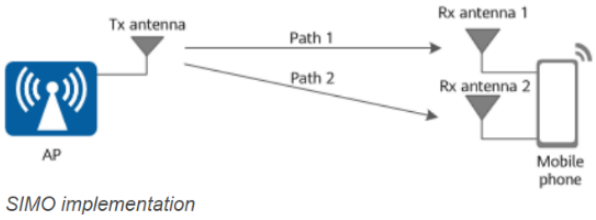

Figure 4 – SIMO implementation [7]

Multiple-input single-output (MISO) is introduced. That is, using two Tx antennas while retaining one Rx antenna. Given the presence of only one Rx antenna, the signals sent from the two Tx antennas must carry the same data and be combined as one signal on the receiver. This mode, referred to as **transmit diversity**, actually delivers the same effect as SIMO.

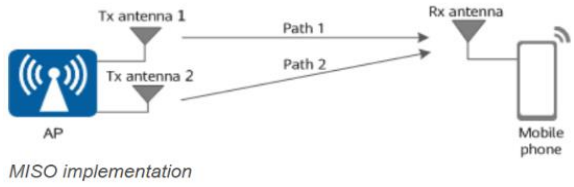

Figure 5 – MISO implementation [7]

The preceding analysis on SIMO and MISO proves that the **transmission capacity depends on the number of Tx and Rx antennas.** Therefore, using two antennas on both the transmitter and receiver can definitely double the rates by transmitting and receiving two signals separately. This mode of using multiple antennas on both the transmitter and receiver is MIMO. [7]

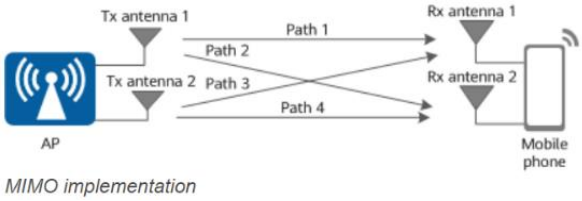

Figure 6 – MIMO implementation [7]

2.1.3 Why MIMO?

MIMO can be used to increase spectral efficiency, channel capacity and link reliability by:

- **Spatial multiplexing:** where multiple independent data streams are transmitted simultaneously over the same frequency band. This dramatically increases the throughput without requiring additional spectrum.
- **Spatial diversity:** Transmit multiple versions of same transmitted signal to improve link reliability. [8]
- **Better coverage:** Using multiple antennas improves signal strength and quality, especially at the edges of a cell or in areas with poor coverage.
- **Beamforming**, a technique supported by MIMO, allows focusing the signal in specific directions, enhancing range and coverage.
- **Overcoming Multipath Effects:** In real-world environments, signals bounce off buildings, trees, and other obstacles, causing **multipath propagation**. MIMO exploits these multiple paths instead of treating them as interference, effectively turning a challenge into an advantage.

Overall, MIMO antennas improve the reliability, performance, and capacity of wireless communication systems by utilizing multiple antennas for simultaneous data transmission and reception.

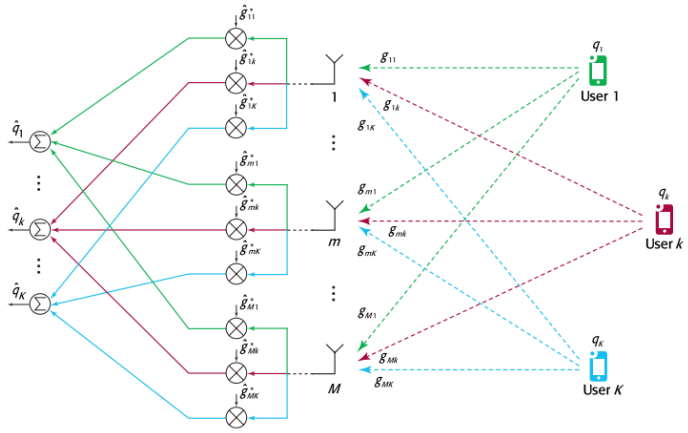

Figure 7 – MIMO as a receiver [20]

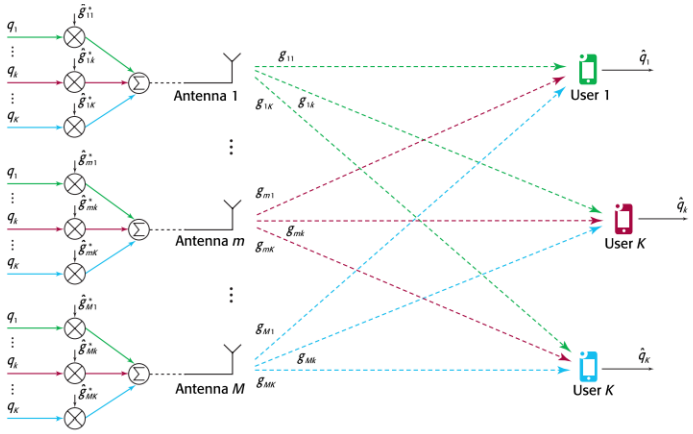

Figure 8 – MIMO as a transmitter [20]

2.1.4 Single-user MIMO

Single-user MIMO focuses all the streams of antenna arrays on a single user. Single-user MIMO technology allows data to transfer simultaneously through more than one data streams to one device. Advantage of SU-MIMO is that there are no interferences and radio channel estimation is easier to achieve. Downside is that it can only serve one device at a time and if transmission matrix is uncorrelated, MIMO cannot be used. [16]

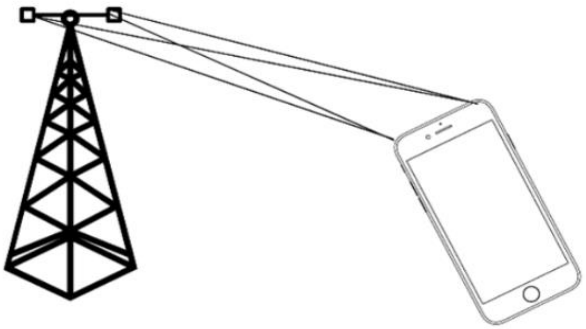

Figure 9 – SU-MIMO [16]

2.1.5 Multi-user MIMO

MU-MIMO is a technology that utilizes more than one antenna to create several connections simultaneously to different devices to improve the network capacity. MU-MIMO, in other words multiple users, multiple input, multiple output, is created to allow multiple users to have access to a base station at the same time. In comparison to standard MIMO, MU-MIMO has multiple data streams to different devices simultaneously.

The advantage of MU-MIMO is that the number of antennas in the devices does not have to be increased. But users' devices should be able to send channel estimation to the base station and base station needs to find optimal UEs for multiuser MIMO, but it has turned out to be challenging. [16]

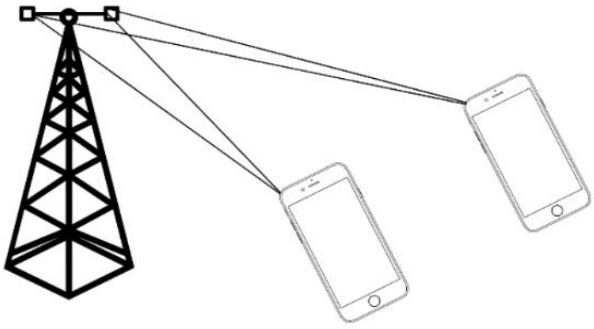

Figure 10 – MU-MIMO [16]

Mathematical Model for Multi-User MIMO System

Consider K independent users in the multi-user MIMO system. We assume that the BS and each MS are equipped with NB and NM antennas, respectively. Fig. 11 shows the uplink channel, known as a multiple access channel (MAC) for K independent users. [17]

Let $x_u \in C^{N_m \times 1}$ and $y_{MAC} \in C^{N_B \times 1}$ denote the transmit signal from the uth user, $u = 1,2,\dots,k$. and the received signal at the BS, respectively.

The channel gain between the uth user MS and BS is represented by $H_u^{UL} \in C^{N_B \times N_M}, u = 1,2,\dots,k$.

The received signal is expressed as

$$y_{MAC} = H_1^{UL}x_1 + H_2^{UL}x_2 + \cdots + H_k^{UL}x_k + z = [H_1^{UL} \quad H_2^{UL} \quad \dots \quad H_k^{UL}]\begin{bmatrix} x_1 \\ x_2 \\ . \\ x_k \end{bmatrix} + z = H^{UL}\begin{bmatrix} x_1 \\ x_2 \\ . \\ x_k \end{bmatrix} + z \qquad (1)$$

where $z \in C^{N_B \times 1}$ is the additive noise in the receiver.

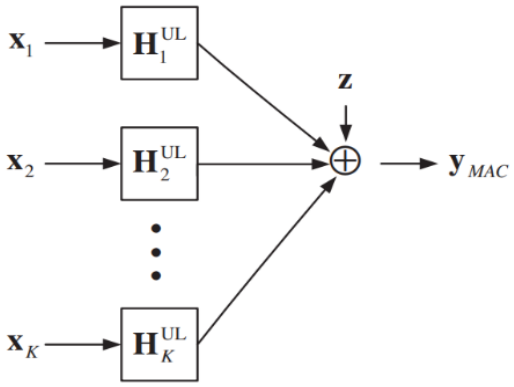

Figure 11 – MU-MIMO uplink channel [17]

Fig. 12 shows the downlink channel, known as a broadcast channel (BC) in which channel $x \in C^{N_B \times 1}$ is the transmit signal from the BS and $y_u \in C^{N_M \times 1}$ is the received signal at the uth user, $u = 1,2,\dots,k$. Let $H_u^{DL} \in C^{N_M \times N_B}$ represent the channel gain between BS and the uth user. In MAC, the received signal at the u th user is expressed as

$$y_u = H_u^{DL}x + z_u\,, \qquad u = 1,2,\dots,k. \qquad (2)$$

where $z_u \in C^{N_M \times 1}$ is the additive noise at the u th user. Representing all user signals by a single vector, the overall system can be represented as

$$\begin{bmatrix} y_1 \\ y_2 \\ . \\ y_k \end{bmatrix} = \begin{bmatrix} H_1^{DL} \\ H_2^{DL} \\ . \\ H_k^{DL} \end{bmatrix} x + \begin{bmatrix} z_1 \\ z_2 \\ . \\ z_k \end{bmatrix} \qquad (3)$$

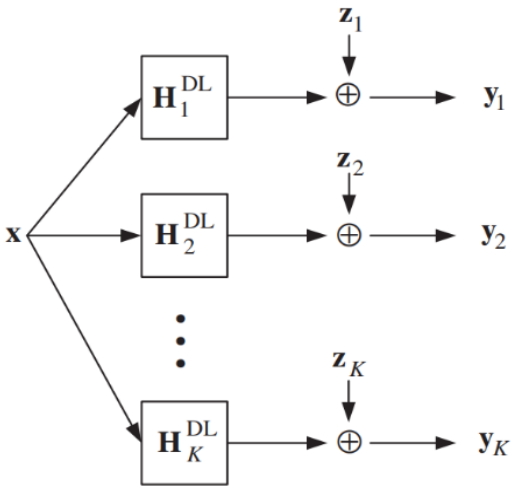

Figure 12 – MU-MIMO downlink channel [17]

2.1.6 What is Massive MIMO?

Massive MIMO or massive multiple-input multiple-output technology is a subset of MU-MIMO technology that involves using a large number of antennas at the transmitter and receiver ends of a wireless communication system or more specifically, at the base stations of cellular networks to improve and increase network performance. [15]

The multiple antennas in a Massive MIMO system work coherently and adaptively to significantly increase throughput, capacity density, and efficiency in a cellular network.

The main concept is to use large antenna arrays at base stations to simultaneously serve many autonomous terminals. The rich and unique propagation signatures of the terminals are exploited with smart processing at the array to achieve superior capacity.

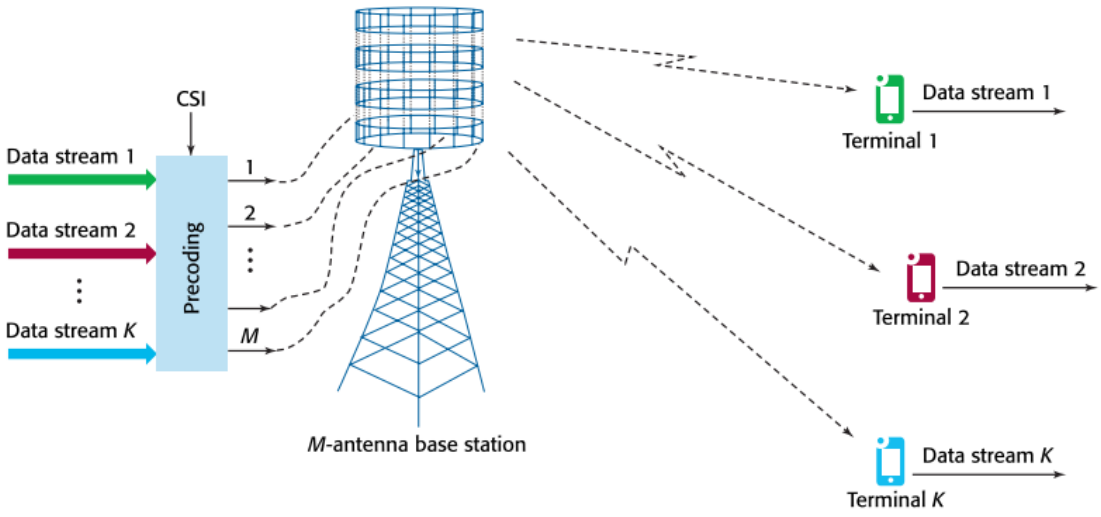

Figure 13 – Massive MIMO Downlink channel [20]

The benefits and applications of Massive MIMO technology:

1. Increases Network Capacity

Massive MIMO increases the capacity of a particular wireless communication network in two ways. First, it enables the deployment of higher frequencies, such as in the case of Sub-6 5G specification. Second, by employing multi-user MIMO, a cellular base station with Massive MIMO capability can send and receive multiple data streams simultaneously from different users using the same frequency resources.

Note that network capacity is determined by the number or amount of total data a particular network can serve to its end-users, as well as by the maximum number of end-users that can be served based on an expected service level.

2. Enhances Network Coverage

Another advantage of Massive MIMO is that it provides high spectral efficiency through the coordination of multiple antennas using simple processing and without intensive power consumption. When used in a 5G cellular network technology, it allows 10 times more spectral and network efficiency compared to fourth-generation networks. Furthermore, when applied in 4G technology, it improves the deep coverage of fourth-generation networks.

Because next-generation cellular network technologies use electromagnetic radiation with higher frequencies or more specifically, frequencies within the upper limits of radio waves and the range of microwaves, the signals they generate travel a short distance. Hence, enhancing network coverage is critical in modern and future cellular technologies.

3. Complements Beamforming

Beamforming technology works by focusing a signal toward a specific direction, rather than broadcasting in all directions, thus resulting in more direct communication between a transmitter and a receiver, more stable and reliable connectivity, and faster data transmission. As a signal processing technique and traffic-signaling system, this technology depends on advanced antenna technologies on both access points and end-user devices.

The large number of antennas in a Massive MIMO system enables three-dimensional beamforming in which a single beam of signal-bearing electromagnetic radiation travels through vertical and horizontal directions. The process increases data transmission rates further while reaching people in elevated areas such as buildings and those in moving vehicles.

4. Enables Next-Gen Technologies

Massive MIMO is an essential component of 5G technology. For example, in Sub-6 5G specification, it allows the utilization of frequencies within the sub-6 GHz range. Moreover, in mm-Wave 5G specification, this technology increases frequency reach to expand network coverage, optimizes the propagation of signal-bearing electromagnetic radiation, and allows true multi-user wireless communication within a defined area.

However, although it is a key enabling technology for 5G and future cellular network technologies, it has been used for improving and repurposing the capabilities of existing 4G systems, especially LTE Advanced networks. The integration of Massive MIMO in existing 4G networks could improve further network performance. [15]

2.1.7 Development of MIMO technology (3G to 5G)

- **In 3G:**

  The MIMO technology was first introduced in the mobile networks in the HSPA Evolution (HSPA+) enhancement as part of the 3G network evolution. However, the uptake of MIMO was more noticeable when 4G LTE (Long Term Evolution) networks were launched as they supported MIMO from the first LTE release (3GPP Release 8). [27]

- **In LTE (4G):**

  4G networks have seen multiple enhancements to the MIMO configuration to improve data rates and signal quality.

The original 4G LTE networks use MIMO configurations of 4×4 in the downlink and 2×2 in the uplink as per 3GPP Release 8; LTE Advanced and LTE Advanced Pro enhancements use 8×8 MIMO configuration in the downlink and 4×4 in the uplink. [18]

LTE Technology	3GPP Release	Antenna Configuration (MIMO)
LTE	Release 8	4 x 4 Downlink 2 x 2 Uplink
LTE Advanced	Release 10	8 x 8 Downlink 4 x 4 Uplink
LTE Advanced Pro	Release 13	8 x 8 Downlink 4 x 4 Uplink

Figure 14 – Antenna configuration for the LTE technologies [18]

- **In 5G:**

  With Massive MIMO, the antenna configuration gets a lot bigger and 64 x 64 is already possible in 5G base stations from key network vendors. However, an antenna configuration of 256 x 256 is also possible.

2.2 Beamforming

2.2.1 What is beamforming?

Beamforming is a signal processing operation used with antenna arrays to create a *spatial* filter; it filters out signals from all directions except the desired direction(s). Beamforming can be used to increase SNR of desired signals, null out interferers, shape beam patterns, or even transmit/receive multiple data streams at the same time and frequency.

As part of beamforming, we use weights (a.k.a. coefficients) applied to each element of an array, either digitally or in analogy circuitry. We manipulate the weights to form the beam(s) of the array, hence the name beamforming! We can steer these beams (and nulls) extremely fast; much faster than mechanically gimballed antennas, which can be thought of as an alternative to phased arrays. [4]

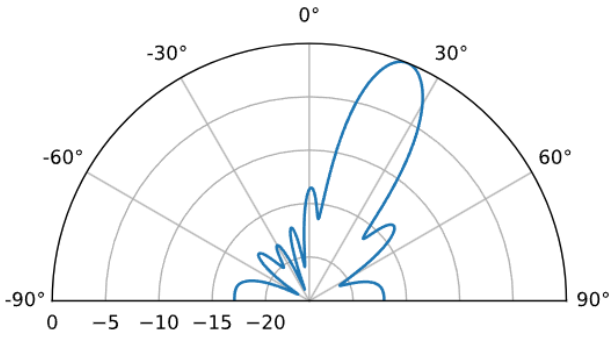

Figure 15 – Beamforming Example [4]

2.2.2 Beamforming in MIMO systems?

Beamforming is the most commonly used method by a new generation of smart antennas. In this method, an array of antennas is used to “steer” or transmit radio signals in a specific direction, rather than simply broadcasting energy/signals in all directions inside the sector. In this method, multiple smaller antennas control the direction of the combined transmitted signal by appropriately weighing the magnitude and phase of each of the smaller antenna signals. In this technique, the phase and amplitude of the transmitted signal of each component antenna are adjusted as needed, resulting in a constructive or destructive effect, concentrating the total transmitted signal into a targeted beam. [21]

The method by which beamforming is achieved is growing increasingly more sophisticated, partly thanks to contemporary **Massive MIMO Antennas**. MIMO technology lets radio signals to be sent and received using several antennas. Massive MIMO antennas have many more component antennas that allow them to transmit radio signals more efficiently. This allows for very high data rates.

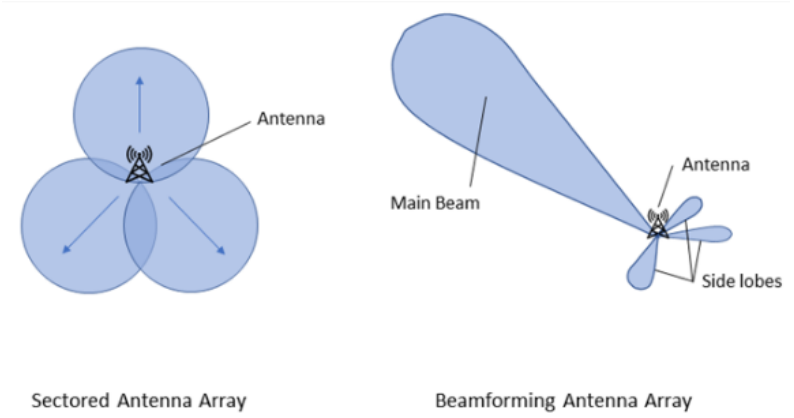

Figure 16 – Beamforming antenna array Vs Sectored antenna array [21]

Beamforming in MIMO systems uses multiple antenna elements to control the direction of a signal. By changing the phase of the individual signals in an antenna array the beam can be formed at the desired angle. The signal can then be directed in the desired direction θ as depicted in Fig. 17.

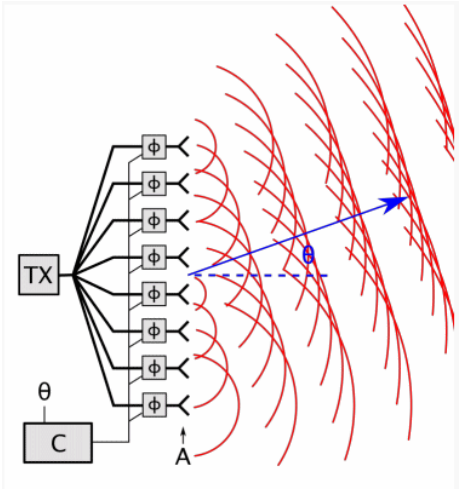

Figure 17 – Beamforming array [21]

2.2.3 Benefits of beamforming

- **Overcoming challenges in 5G networks**

As 5G networks continue their global rollout, enabling smartphones, connected devices, and wide-area networking applications, beamforming emerges as a critical core technology. The high-frequency millimeter wave spectrum used in 5G communications is susceptible to disruptions from obstacles like walls and other barriers, posing challenges for reliable connectivity. Beamforming plays a crucial role in mitigating these challenges by allowing transmitters to focus their signals in specific directions towards mobile devices, vehicles, or Internet of Things (IoT) devices. This directional transmission not only enhances signal strength but also improves the overall reliability of 5G connections.

- **Enhanced performance, Efficiency and capacity**

Moreover, beamforming will be a key enabler for Massive MIMO technology in 5G networks, which, in conjunction with beamforming, can direct beams both horizontally and vertically towards user devices. This precise beam steering capability not only maximizes throughput but also optimizes spectral efficiency, leading to improved network performance and capacity.

- **Enhanced reliability, speed and latency**

By precisely shaping and directing the radio frequency (RF) signals, beamforming overcomes the inherent limitations of mm-Wave frequencies, ensuring reliable, high-speed, and low-latency connectivity for 5G applications. As a pivotal technology, beamforming is poised to play a vital role in unlocking the full potential of 5G networks, enabling seamless and efficient wireless communications for a vast array of devices and use case. [2]

Overall advantages and Challenges:

Advantages: Beamforming improves signal quality, extends coverage range, mitigates interference, and enhances spectral efficiency, leading to better overall performance.

Challenges: Beamforming systems require accurate channel estimation, synchronization, and coordination, while also facing limitations in terms of hardware complexity and power consumption.

2.2.4 Beamforming techniques

Beamforming approaches can be broken down into three categories, namely: conventional, adaptive, and blind. [4]

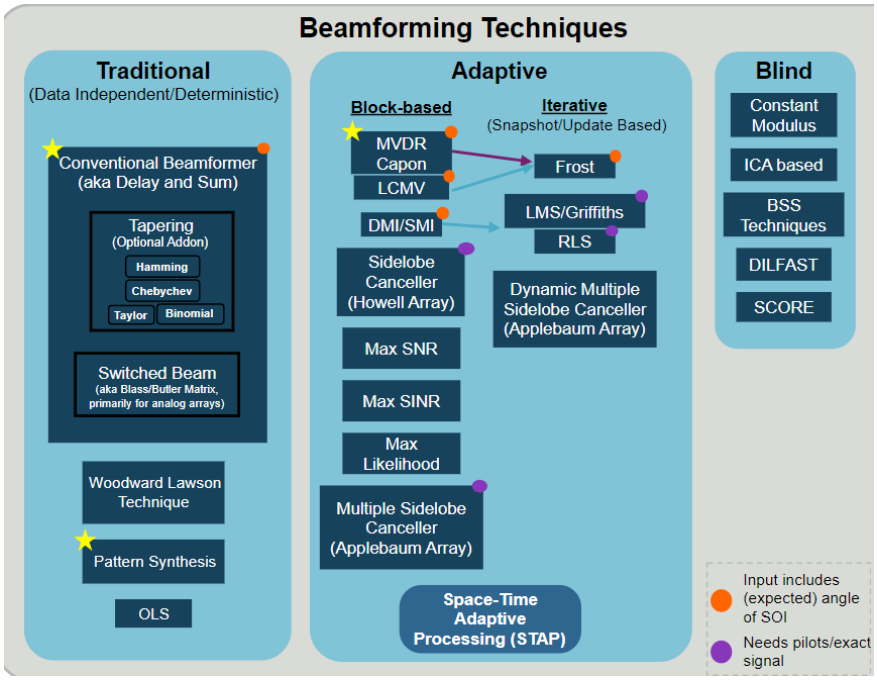

Figure 18 – Beamforming Techniques [4]

Conventional beamforming is most useful when you already know the direction of arrival of the signal of interest, and the beamforming process involves choosing weights to maximize the array gain in that direction. This can be used on both the receive or transmit side of a communication system.

Adaptive beamforming, on the other hand, typically involves adjusting the weights based on the beamformer's input, to optimize some criteria (e.g., nulling out an interferer, having multiple main beams, etc.). Due to the closed loop and adaptive nature, adaptive beamforming is typically just used on the receive side, so the "beamformer's input" is simply your received signal, and adaptive beamforming involves adjusting the weights based on the statistics of that received data. [4]

Types of beamforming

1- Analog beamforming

- Phase shifters or variable attenuators are used at each antenna element to adjust the phase and amplitude of signals.
- Signals are combined into a single beam.
- Lower power consumption but limited flexibility.
- Difficult to support multiple beams or dynamic adaptation.
- Uses single RF chain.

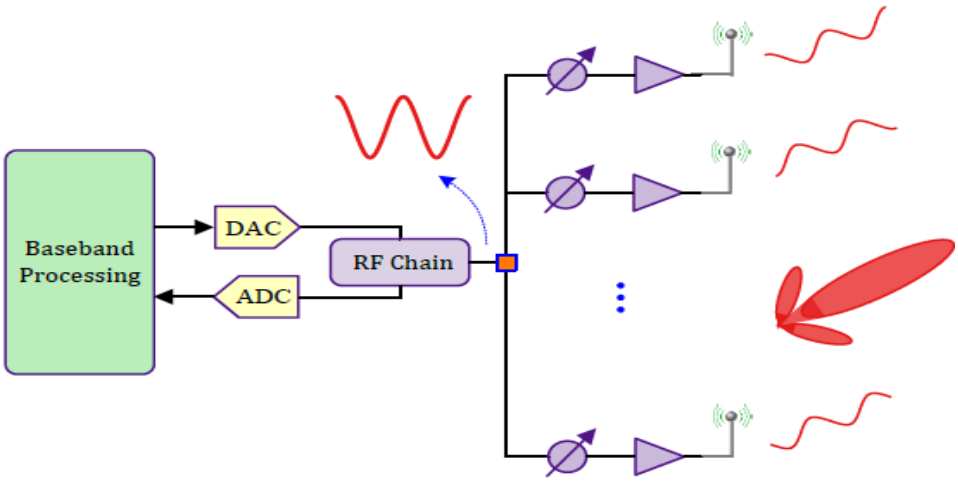

Figure 19 – Analog Beamforming Receiver [10]

2- Digital beamforming

- Signals from all antenna elements are processed digitally using algorithms to form and steer beams.
- Can form multiple beams simultaneously.
- Supports dynamic beam steering and adaptation.

- Requires high computational power and energy due to the need for multiple ADCs/DACs.
- Uses multiple RF chains.

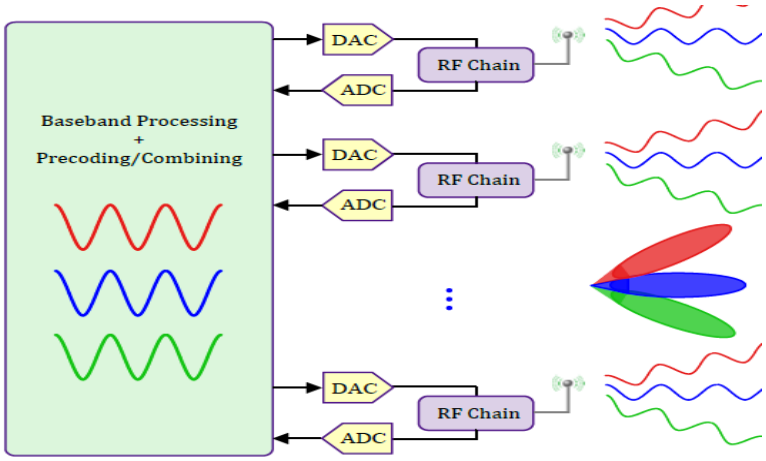

Figure 20 – Digital Beamforming Receiver [10]

3- Hybrid beamforming

- Hybrid beamforming combines analog and digital beamforming, striking a balance between flexibility and cost.
- A smaller number of digital signal chains are used, with analog phase shifters applied for additional processing.
- The signal is split into multiple analog beams and then processed digitally.
- Reduces the cost and power consumption of fully digital systems. [10]

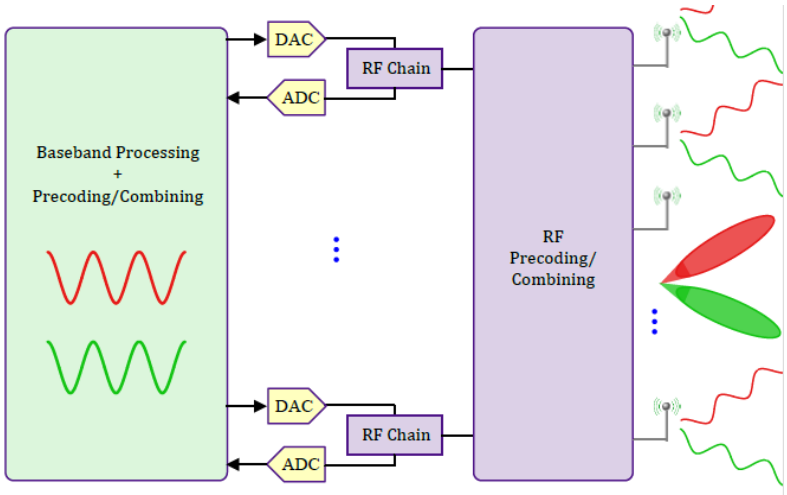

Figure 21 – Hybrid Beamforming Receiver [10]

2.3 Digital beamforming

In digital beamforming, each antenna has its own RF chain, and the signal processing is done in the digital domain. This allows more control over the beam patterns and supports multiple beams and users.

Key Advantages of Digital Beamforming:

- Increased Control: Digital beamforming allows for precise control over beam patterns. The digital processing can adaptively change the beam shape and direction based on real-time conditions or user locations.
- Multiple Beams: Digital beamforming can create multiple independent beams simultaneously. This capability supports MIMO (Multiple Input Multiple Output), enabling the base station to serve multiple users at different angles with distinct beams.
- Flexibility: The ability to adjust both phase and amplitude means that digital beamforming can optimize performance for different environments and user requirements, making it much more versatile than analog beamforming.

Digital beamforming represents a significant advancement over analog beamforming by enabling enhanced flexibility and capacity to manage complex communication scenarios, particularly in modern wireless networks like 5G.

2.3.1 Phased arrays:

A phased array is an electronically scanned array, a computer-controlled array of antennas which creates a beam of radio waves that can be electronically steered to point in different directions without moving the antennas. The general theory of an electromagnetic phased array also finds applications in ultrasonic and medical imaging application (phased array ultrasonics) and in optics optical phased array.

In a simple antenna array, the radio frequency current from the transmitter is fed to multiple individual antenna elements with the proper phase relationship so that the radio waves from the separate elements combine (superpose) to form beams, to increase power radiated in desired directions and suppress radiation in undesired directions. [3]

Phased arrays are introduced as versatile antenna systems capable of shaping and scanning radiation patterns by adjusting the spacing and excitation of multiple elements. Unlike traditional mechanically steered aperture antennas, phased arrays use electronic phase adjustments to achieve rapid and precise beam scanning, eliminating the need for mechanical movement. This approach also supports multiple simultaneous beams and conformal geometries, making phased arrays suitable for applications like radar.

To excite an antenna, we apply alternating current on its terminals as in Fig. 22.

To steer a beam in a certain direction using phased array we excite the antennas with the same current amplitude but different phase β. By changing the phase shifts between the elements, we change the positions of the constructive (Addition of electric fields vectors) and distractive (cancellation of electric fields vectors) interferences which changes the directions of beam maxima and minima. Phased arrays increases the overall directivity (narrower beam) and gain.

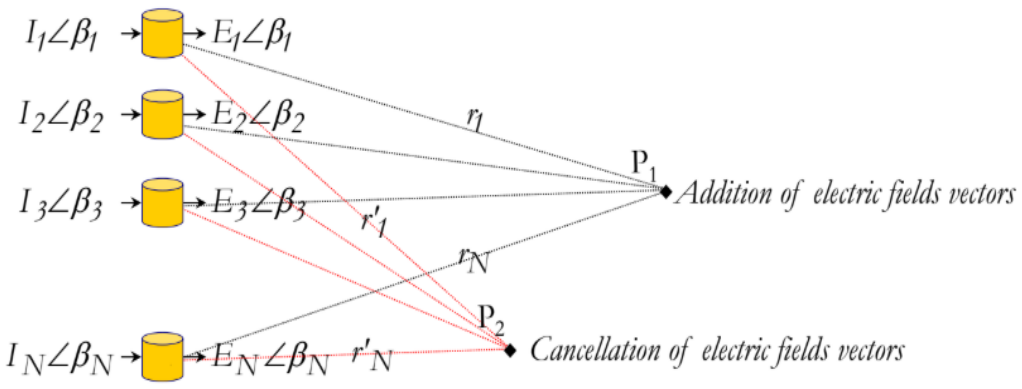

Figure 22 – Antenna array model [12]

Types of phased arrays:

- **Uniform phased array:**
  Here, both the amplitude and phase of the current vary across the array elements. This approach allows for side lobe suppression (also known as gain tapering) and provides more precise beamforming, enhancing the array's overall performance.
- **Non uniform phased array:**

  Here, both the amplitude and phase of the current vary across the array elements. This approach allows for side lobe suppression (also known as gain tapering) and provides more precise beamforming, enhancing the array's overall performance.

The factors affecting phased arrays:

1- Number of elements:

As the number of elements increase the gain, directivity and control increase but this also increases cost, power consumption and complexity of the signal processing.

2- Element spacing:

As the spacing increase the beam becomes more sharper but grading lobes appear in the beam pattern.

3- Array geometry

The way that we construct the array and position the antennas. The common geometry patterns are linear, planar, and circular arrays.

Starting with one isotropic antenna that radiates a sinusoidal wave in all directions, Fig. 23 shows the wave front of this radiation.

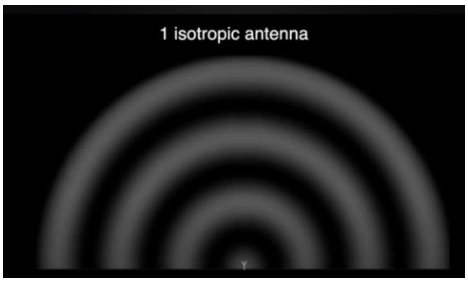

Figure 23 – Single isotropic antenna radiation pattern [11]

Two identical isotropic antennas with spacing of $d = \frac{\lambda}{2}$

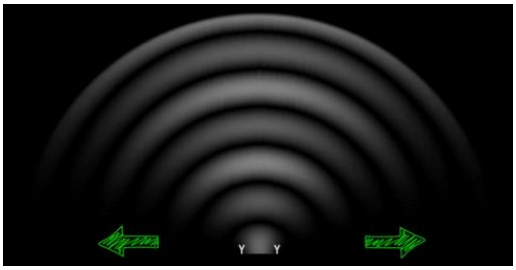

Figure 24 – Two isotropic antennas radiation pattern [11]

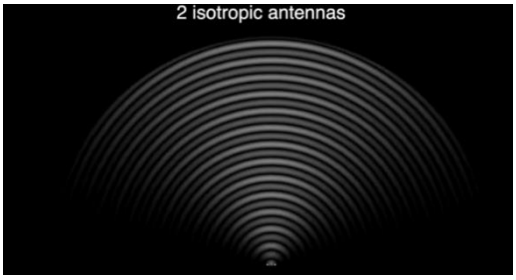

Figure 25 – Two isotropic antennas radiation pattern (zoomed out) [11]

Fig. 24 and 25 show destructive interference in the right and left sides and constructive interference on the top side resulting in more directive output beam than the case with single antenna.

For sharper beam

1- Increase the distance between two antennas (ex, spacing = λ) leads to sharper beam but grating lobes appears as we see in Fig. 26. Grating lobes are an unwanted lobe that have gain comparable with the gain of the main lobe (power loss).

2- Increase the number of antennas with the same initial spacing of $\frac{\lambda}{2}$ which increases the directivity as we see in Fig. 27. Side lobes appear but their power is much less than that of main lobe.

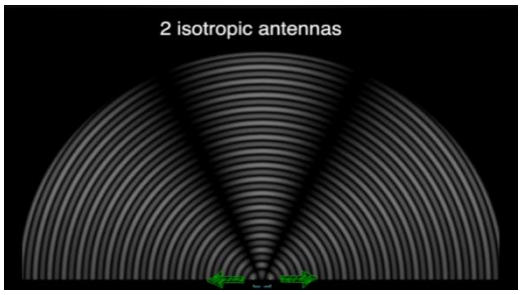

Figure 26 – Two isotropic antennas with spacing λ radiation pattern [11]

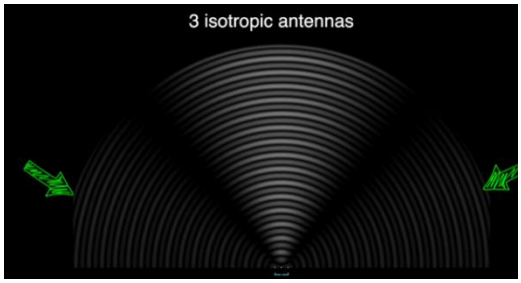

Figure 27 – Three isotropic antennas radiation pattern [11]

For steering the phase of one of the two antennas is changed (ex, phase shift = 0.5 lambda) which produces the result in Fig. 28.

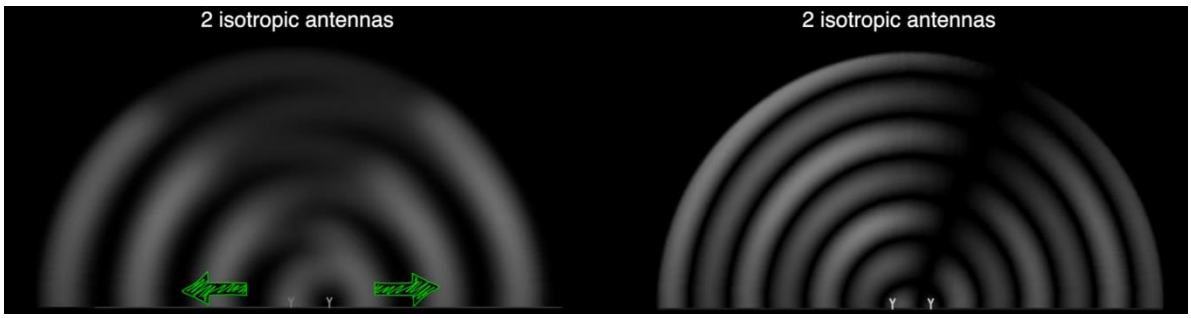

Figure 28 – Two isotropic antennas with phase shift radiation pattern [11]

By comparing with the previous results in Fig. 24 and 25 with Fig. 28, applying a phase shift causes a rotation in the main beam.

8-Element array

For an 8-Element antenna array, more clear results are depicted in Fig. 29. Increasing the number of elements results in a sharper beam.

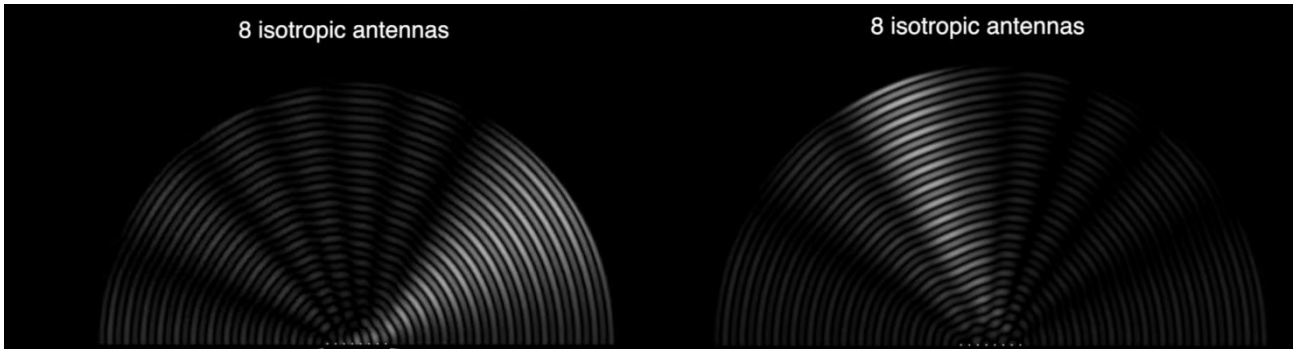

Figure 29 – Eight isotropic antennas with beam steering [11]

2.3.2 Steering Vector

Consider a 1D uniformly spaced array:

In this example, the signal comes from the right side, so it hits the right element first as shown in Fig. 30.

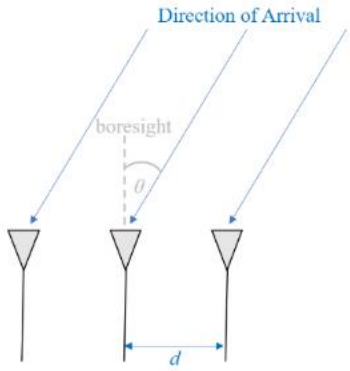

Figure 30 – ULA [4]

To calculate the delay between when the signal hits that first element and when it reaches the next element the following trig problem is formed.

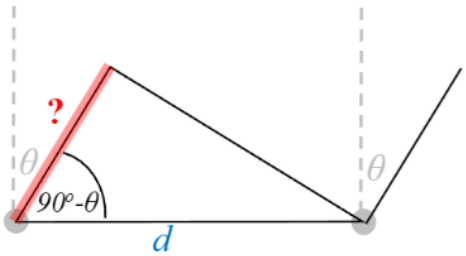

Figure 31 – Delay calculation [4]

The segment highlighted in red in Fig. 31 is the distance the signal has to travel after it has reached the first element, before it hits the next one.

$$adjacent = dsin(\theta) \tag{4}$$

To connect this trig and speed of light math to the signal processing world. The transmit signal at baseband is $x(t)$ and it's being transmitting at some carrier, f_c , so the transmit signal is

$$x(t)e^{j2\pi f_c t} \tag{5}$$

We'll use d_m to refer to antenna spacing in meters. Let's say this signal hits the first element at time $t = 0$, which means it hits the next element after

$$\frac{d_m \sin(\theta)}{c} \tag{6}$$

This means the 2nd element receives:

$$x(t - \Delta t)e^{j2\pi f_c(t-\Delta t)} \tag{7}$$

$$\Delta t = \frac{d_m sin(\theta)}{c} \tag{8}$$

When **down conversion** is done at the receiver, it's essentially multiplying the received signal by the carrier but in the reverse direction. After down conversion the receiver sees:

$$x(t - \Delta t)e^{j2\pi f_c(t-\Delta t)}e^{-j2\pi f_c t} = x(t - \Delta t)e^{-j2\pi f_c \Delta t} \tag{9}$$

Now to simplify this even further; when a signal is sampled it can be modeled by substituting t for nT where T is sample period and n is just 0, 1, 2, 3…

Substituting this in we get

$$x(nT - \Delta t)e^{-j2\pi f_c \Delta t} \tag{10}$$

nT is so much greater than Δt so we can get rid of the first Δt term

$$x(nT)e^{-j2\pi f_c \Delta t} \tag{11}$$

If the sample rate ever gets fast enough to approach the speed of light over a tiny distance, we can revisit this, but the sample rate only needs to be double the bandwidth of the signal of interest.

Eq. (11) can be represented in discrete terms as the following with Δt plugged in:

$$x[n]e^{-j2\pi f_c \Delta t} = x[n]e^{-j2\pi f_c(\frac{d_m \sin(\theta)}{c})} \tag{12}$$

$$\lambda = \frac{c}{f_c} \tag{13}$$

$$x[n]e^{-j2\pi(\frac{d_m \sin(\theta)}{\lambda})} \tag{14}$$

In beamforming, d (the distance between adjacent elements) is represented as a fraction of wavelength (instead of meters). The most common value chosen for d during the array design process is half the wavelength.

d (without the subscript m) represents normalized distance

$$d = \frac{d_m}{\lambda} \tag{15}$$

We can simplify (14) to:

$$x[n]e^{-j2\pi dsin(\theta)} \tag{16}$$

Eq. (16) is specific to adjacent elements, for the signal received by the k'th element we just need to multiply d times k:

$$x[n]e^{-j2\pi dksin(\theta)} \tag{17}$$

The matrix denoted by s is the called the "steering vector"

$$s = \begin{bmatrix} 1 \\ e^{-j2\pi d(1)sin(\theta)} \\ e^{-j2\pi d(2)sin(\theta)} \\ \vdots \\ e^{-j2\pi d(N_r-1)sin(\theta)} \end{bmatrix} \tag{18}$$

To get the received signals at the different N_r antenna elements we multiply the transmitted signal $x[n]$ by the steering vector s

$$X = s \times x[n] \tag{19}$$

$x[n]$: The transmitted signal vector of size $1 \times N$ (N is the number of samples per signal).

s: The steering vector of size $N_r \times 1$.

X: The received signals at the N_r elements ($N_r \times N$). [4]

2.3.3 Delay and sum Beamforming

On the top of Fig. 32, we have a scenario where a desired signal is received and the appropriate time delays (t_1, t_2) are applied such that the signals constructively interfere and the array produces a high response. On the bottom, we have an undesired signal and no time delays are applied, there is no constructive interference and the array does not have a high response to this signal. [5]

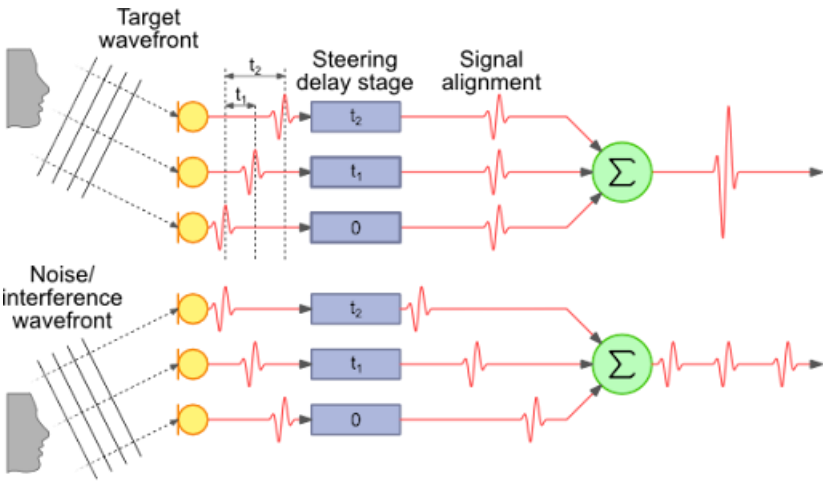

Figure 32 – Delay and sum beamforming [5]

The Delay-and-Sum Beamformer applies a time delay to the incoming signal from each element and sums the output together. If we get the time delays correct, we will have a single high output signal. We can then use the time delays that produced this signal to determine the angle of its arrival.

Our weights vector w needs to be a 1D array for a uniform linear array. With delay and sum beamforming we leave the magnitude of the weights at 1, and adjust the phases so that the signal constructively adds up in the direction of our desired signal, which we will refer to as θ. In delay and sum it turns out that it has the exact same math we did above in the steering vector so our weights are our steering vector. [4]

$$w = e^{-j2\pi dksin(\theta)} \quad (20)$$

$$Y = w^H.X \quad (21)$$

w: weights vector of size $N_r \times 1$

Y: The beamformed signal $(1 \times N)$.

DOA

But how do we know the angle of interest theta? We must start by performing DOA, which involves scanning through (sampling) all directions of arrival from -π to +π (-180 to +180 degrees). At each direction we calculate the weights using a beamformer. Applying the weights to our signal will give us a 1D array of samples, as if we received it with 1 directional antenna. We can then calculate the power in the signal by taking the variance, and repeat for every angle in our scan. We plot the results and look at it with our human eyes, but what most RF DSP does is find the angle of maximum power (with a peak-finding algorithm) and call it the **DOA estimate**. [4]

2.2.4 MVDR/Capon Beamformer

A beamformer that is slightly more complicated than the conventional/delay-and-sum technique, but tends to perform much better is called the Minimum Variance Distortion Less Response (MVDR) or Capon Beamformer. We know that variance of a signal corresponds to how much power is in the signal. The idea behind MVDR is to keep the signal at the angle of interest at a fixed gain of 1 (0 dB), while minimizing the total variance/power of the resulting beamformed signal. If our signal of interest is kept fixed then minimizing the total power means minimizing interferers and noise as much as possible. It is often referred to as a "statistically optimal" beamformer.

Let's recall (21), we process the received signal **X** with a matrix multiplication.

We can impose a constraint on ***w*** such that the resulting Array Output ***Y*** is exactly equal to the original signal $\boldsymbol{f}$ (previously $x[n]$).

$$Y = f \tag{22}$$

Since the recovered signal is exactly equal to the original (*i.e. no distortion*) this is called the **Distortion less Response**. Now, if ***w*** is actually equal to the true signal steering vector $\mathbf{s}(\Theta_{\mathbf{s}})$ then their inner product will be equal to 1. We can reformulate this constraint with the following expression.

$$s^H.w = 1 \quad or \quad w^H.s = 1 \tag{23}$$

This is our **Distortion less Constraint** for our Beamformer ***w***. This essentially does two things.

First, it forces our Beamformer to be *steered* in the direction of the true signal by forcing the energy of the Beamformer response to be high in this direction.

Second, it forces the energy of the response to be low in all other directions, this is sometimes called ***steering a null*** in all other directions. **The drawback of this, is that our Beamformer is intended to work for a single signal, since we are only considering a single signal in our model**.

Now let's turn our attention to the output of this Beamformer so that we can construct the objective function.

Distortion less Response of a Noisy Signal

Let's incorporate the Distortion less Response into the General Beamformer and inspect its response with a noisy signal

$$X_n = X + n \tag{24}$$

where **n** is zero mean Additive Gaussian White Noise (AWGN).

The output of the beamformer using a weight vector w is given by:

$$\begin{aligned} y &= w^H . X_n \\ &= w^H . (X + n) \\ &= w^H . X + w^H . n \\ &= f + w^H . n \\ Y &= f + y_n \end{aligned} \tag{25}$$

We have the original signal $\boldsymbol{x[n]}$ plus the processed noise term y_n, this extra noise term will cause errors in our estimates. We need to find a way to minimize this noise in order to retain an accurate estimate. The power of the noise is characterized by its variance, so a proper way to state our goal would be to minimize the variance of the noise. But there's a problem here, we don't get our signal and noise separately, we get them together in a package deal. So, we really need to minimize the variance of the total response ***Y*** (*i.e. the term we have access to*) so that our estimate is accurate. Now let's piece this together, our response is assumed to be distortion less without the presence of noise, so we will minimize the variance of the distortion less response.

Since we are minimizing the variance of the Distortion less response, let's see what this variance term actually looks like. Below, we simplify the variance of the Distortion less Response Y.

$$\begin{aligned} var(Y) &= E[|Y|^2] + E[Y]^2 \\ &= E[|Y|^2] + E[f + y_n]^2 \\ &= E[|Y|^2] + (E[f] + E[y_n])^2 \\ &= E[|Y|^2] + 0 \\ &= E(|w^H . X_n|^2) \\ &= E[(w^H . X_n) . (w^H . X_n)^H] \\ &= E[w^H . X_n . w . X_n^H] \\ &= w^H . E[X_n . X_n^H] . w \end{aligned}$$

$$var(Y) = w^H . R . w \tag{26}$$

We have used the fact that f and y_n are zero mean.

R is an $N_r \times N_r$ sample correlation matrix.

w is the Beamformer weighting that will produce the recovered signal.

This expression is our objective function that we will minimize subject to the Distortion less Constraint.

The Optimization Problem:

Minimize $\boldsymbol{w^H . R . w}$ (minimize the total power of the signal)

Subject to: $\boldsymbol{w^H . s = 1}$ (no distortion and the received signal is exactly like the original one)

Lagrangian Method - Introduce a Lagrange multiplier λ and form the Lagrangian:

$$L(\mathrm{w},\lambda) = w^H.R.w - \lambda.(\mathrm{w^H}.\mathrm{s} - 1) \tag{27}$$

Solving the Optimization - Differentiating the Lagrangian with respect to the w^H and setting the derivative to zero, we obtain:

$$\frac{\partial L}{\partial w^*} = 2R.w - \lambda.\mathrm{s} = 0$$
$$w = \lambda.s.R^{-1}$$
$$w = \frac{\lambda.s}{R} \tag{28}$$

To solve for λ, apply the constraint $\boldsymbol{w^H.s = 1}$

$$(\lambda.s^H.R^{-1}).s = 1$$
$$\lambda = \frac{1}{s^H.R^{-1}.s} \tag{29}$$

substituting (29) in (28) we get

$$\boldsymbol{w_{mvdr} = \frac{R^{-1}.s}{s^H.R^{-1}.s}} \tag{30}$$

If we already know the direction of the signal of interest, and that direction does not change, we only have to calculate the weights once and simply use them to receive our signal of interest. Although even if the direction doesn't change, we benefit from recalculating these weights periodically, to account for changes in the interference/noise, which is why these non-conventional digital beamformers are referred to as "adaptive" beamformers; they use the received signal information to calculate the weights.

We can perform beamforming using MVDR by calculating these weights and applying them to the, just like we did in the conventional method, the only difference is how the weights are calculated. [5]

DOA

We simply repeat the MVDR calculation while scanning through all angles of interest. At each angle we calculate the MVDR weights, then apply them to the received signal, then calculate the power in that signal. The angle that gives us the highest power is our DOA estimate.

2.3.5 LCMV Beamformer

Linearly Constrained Minimum Variance (LCMV) beamformer is used if we have more than one SOI (signal of interest). It is a generalization of MVDR, where we specify the desired response for multiple directions. [26]

LCMV Beamformer determines the optimal weight vector that minimizes the output power while satisfying one or more linear equality constraints. The optimization problem for this beamformer is given by,

$$\min \boldsymbol{w^H R w} \ \ subject\ to\ k\ linear\ constraints\ \boldsymbol{C^H w = f}$$

The constraint matrix C contains k steering vectors and f is the gain vector corresponding to each steering vector contained in the matrix C. The solution to this optimization problem comes out to be,

$$w_{lcmv} = R^{-1}C[C^H R^{-1} C]^{-1} f \tag{31}$$

R is an $N \times N$ sample correlation matrix.

C is a $N_r \times n$ matrix comprising of steering vectors at the AOAs of (n) SOIs and interferers.

f the desired response vector. The vector f for a particular row takes the value of 0 when the corresponding steering vector is to be nulled, and takes a value of 1 when we want a beam pointed at it. For example, if we have two sources of interest and two sources of interference, we can set f = [1,1,0,0].

The LCMV beamformer is a powerful tool that can be used to suppress interference and noise from multiple directions while simultaneously enhancing the signal of interest from multiple directions.

The catch is that the total number of nulls and beams (size of the vector f) you can form simultaneously is limited by the size of the array (the number of elements).

Furthermore, you need to craft the steering vector for each of the SOIs (signals of interest) and interferers, which isn't always readily available in practical applications.

When estimates are used instead, the performance of the LCMV beamformer can degrade. It is for this reason that we prefer to steer nulls using the spatial covariance matrix R (based on statistics of the received signal), instead of "hardcoding" nulls by estimating the AoA of the interferer (which could have error) and crafting the steering vector in that direction, with a 0 added to f. [4]

2.3.6 Subspace Techniques

We will now talk about a different kind of beamformers, the “sub-space” methods. These involve dividing the signal subspace and noise subspace, which means we must estimate how many signals are being received by the array for good results.

2.3.6.1 MUSIC Algorithm

Multiple Signal Classification (MUSIC) is a high-resolution algorithm for estimating the Directions of Arrival (DOAs). The MUSIC algorithm is based on the decomposition of the covariance matrix of signals received from antenna array.

Eigen decomposition is an operation that breaks a matrix down into its eigenvalues and eigenvectors to help you better understand its properties.

After calculating the eigenvectors of the covariance matrix, we split the eigenvectors into two groups: signal sub-space and noise-subspace, then project steering vectors into the noise sub-space and steer for nulls. [13]

MUSIC algorithm uses the eigenvectors decomposition and eigenvalues of the covariance matrix of the antenna array for estimating directions-of-arrival of sources based on the properties of the signal and noise subspaces.

For this, the initial hypothesis is that the covariance matrix R is not singular. This assumption physically means that sources are totally uncorrelated between them. MUSIC algorithm assumes that signal and noise subspaces are orthogonal. The signal subspace E consists of phase shift vectors between antennas depending on the angle of arrival. All orthogonal vectors to E_s constitute a subspace E_N called noise subspace.

MUSIC algorithm being based on the properties of signal and noise subspaces, vectors derived from E_s generate a signal subspace collinear with steering vectors of sources $a(\theta_k)$ and vectors derived from E_N generate a noise subspace **orthogonal** to the steering vectors of these sources.

It follows that:

$$\boldsymbol{E_N^H . a(\theta_k) = 0} \;\; for\; k \; = 1, 2, \dots, K \tag{32}$$

For determining various directions-of-arrival, it is necessary to diagonalize the covariance matrix of the data, identify the signal and noise space, and project onto the noise space. The principle is to project all possible directional vectors (averaging over several vectors) on the noise subspace and retain only those minimizing this projection, resulting in the following discriminant function:

$$d^2 = a(\theta)^H E_N E_N^H a(\theta) = 0 \tag{33}$$

$$(E_N = [e_1, e_2, \dots, e_{M-K}]) \tag{34}$$

whose zeros are the directions-of-arrival.

$C = E_N . E_N^H$ is the projection matrix and $a(\theta)^H E_N E_N^H a(\theta)$ is the projection of the vector $a(\theta)$ on the noise subspace (the projection is zero at the true DOA).

Estimating directions-of-arrival of signals is equivalent to look for maximum values of the MUSIC pseudo-spectrum $P(\theta)$:

$$P_{MUSIC}(\theta) = \frac{1}{A(\theta) E_n E_n^H A(\theta)} \quad (35)$$

The MUSIC equation that will be used to determine the angle of arrival is the following: [4]

$$\theta = argmax(\frac{1}{s^H V_n V_n^H s}) \quad (36)$$

Where:

V_n is that list of noise sub-space eigenvectors (a 2D matrix).

s is the steering vector

argmax operator finds the angle θ at which the MUSIC spectrum is maximized.

V_n is found by first calculating the eigenvectors of the covariance matrix R , and then splitting up the vectors based on how many signals we think the array is receiving. The number of signals must be between 1 and $N_r - 1$. If you are designing an array, when you are choosing the number of elements you must have one more than the number of anticipated signals.

Note: V_n does not depend on the steering vector s. [4]

2.3.6.2 ESPRIT Algorithm

ESPRIT (Estimation of Signal Parameters via Rotational Invariance Techniques) was developed by Roy and Kailath in 1989. It is based on the rotational invariance property of the signal space to make a direct estimation of the DOA and obtain the angles of arrival without the calculation of a pseudo-spectrum on the extent of space, nor even the search for roots of a polynomial. This method exploits the property of translational invariance of the antenna array by decomposing the main network into two sub-networks of identical antennas which one can be obtained by a translation of the other as depicted in Fig. 33. [13]

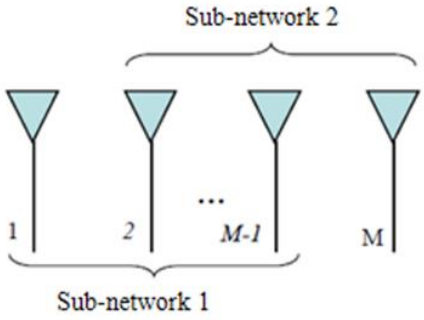

Figure 33 – Decomposing the network into two sub-networks [13]

The main advantage of this method is that it avoids the heavy research of maxima of a pseudo-spectrum or a cost function (therefore a gain calculation) and the simplicity of its implementation. In addition, this technique is less sensitive to noise than MUSIC. By designating $x_1(t)$ and $x_2(t)$ as observation vectors at the outputs of sub-networks 1 and 2, the received signal vector in baseband of the complete network is written as following:

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(2) \end{bmatrix} = \begin{bmatrix} A \\ A.\Phi \end{bmatrix}.S_k(t) + n(t) \tag{37}$$

$$\Phi = diag[e^{j\frac{2\pi}{\lambda}dsin\theta_1}, e^{j\frac{2\pi}{\lambda}dsin\theta_2}, \dots, e^{j\frac{2\pi}{\lambda}dsin\theta_k}] \tag{38}$$

This relation will allow estimation of the angles of arrival without knowing the expression of the matrix A of sources vectors. It so allows the use of the ESPRIT algorithm to antennas of badly known or unknown geometry.

The correlation matrix of the complete network is given by:

$$R_{xx} = \begin{bmatrix} A \\ A.\Phi \end{bmatrix} R_{ss} \begin{bmatrix} A^H \\ A^H.\Phi^H \end{bmatrix} + \sigma^2 I \tag{39}$$

Where $A = [a(\theta_1), a(\theta_2), \dots, a(\theta_k)]$ is a MxK matrix of the source vectors defined in a sub-network and R_{ss} is the spatial matrix of sources.

The matrix R_{xx} being Hermitian, its eigenvalues are real: $(\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k \geq \lambda_{k+1} = \sigma^2)$. The largest K eigenvalues correspond to the space signal generated by the K sources. The signal subspace E_S is a $M \times K$ matrix composed of K eigenvectors associated with the signal subspace. The signal subspace E_S of the whole network can be decomposed into two subspaces E_1 and E_2 which are the $(M-1) \times K$ matrices whose columns are composed of K eigenvectors corresponding to eigenvalues of the covariance matrices of the sub-networks 1 and 2.

These two matrices E_1 and E_2 are related by the following relation of invertible linear transformation:

$$E_s = \begin{bmatrix} E_1 \\ E_2 \end{bmatrix} = \begin{bmatrix} AT \\ A.\Phi T \end{bmatrix} \quad (40)$$

$$T = R_{11}^{-1}.R_{21} \quad (41)$$

$$R_{11} = \frac{1}{N} X_1 X_1^H \quad (42)$$

$$R_{21} = \frac{1}{N} X_2 X_1^H \quad (43)$$

R_{11} and R_{21} are the covariance matrices between the two sub-networks of antennas.

$$E_2 = ATT^{-1}\Phi T = E_1\Psi \quad (44)$$

Where the rotation matrix is a $K \times K$ matrix

$$\Psi = E_2/E_1 \quad (45)$$

The eigenvalues of Φ and Ψ are common and are expressed by $\lambda_i = e^{jkdsin\theta_i}$ with $k = 1, 2, \dots, K$

The angles of arrival are given by

$$\lambda_i = |\lambda_i| e^{jarg(\lambda_i)} \quad (46)$$

$$\theta_i = \sin^{-1}\left(\frac{arg(\lambda_i)}{kd}\right) \quad i = 1, 2, \dots, K \quad (47)$$

$K = \frac{2\pi}{\lambda}$ is the wave number

d is the distance between antennas in meters so

$$Kd = \frac{2\pi d}{\lambda} \quad (48)$$

In practice to determine directions of arrival, it is necessary to:

- Achieve the decomposition in singular value of the data covariance matrix $R_{xx} = \frac{1}{N} XX^H$ where N is the number of observations.
- Estimate the dimension of the signal subspace Es (40).
- Separate the eigenvectors corresponding to the signal subspace and form the matrices of sub-networks E_1 and E_2.
- Estimate the rotation operator Ψ (45).
- Calculate the eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_k$ of the matrix Ψ (46).
- Finally, calculate the AOAs (47).

Ch 3: MATLAB Simulations

For the simulation of the different beamforming techniques, the beamforming process is simulated in the receiving mode by using these following general steps:

1- A baseband signal is defined at the transmitter which is the signal of interest.
2- The direction of arrival (θ) is defined and the array parameters are set (ex, number of antennas, element spacing).
3- The steering vector in section 2.3.2 which depends on the angle of arrival θ is used to model the phase change (delay) at each antenna.
4- The signal is transmitted through an AWGN channel to model a real scenario.
5- At the receiver the weights which depend on the beamformer simulated are generated and multiplied by the received signal (transmitted + AWGN) to generate the beamformed signal.
6- For investigating the DOA, a scan is performed on different thetas by making the previous steps at each scan angle and a peak finding algorithm is used to determine the angle of arrival.

3.1 Delay and sum

For a signal with AOA of $30°$ and a ULA with $N_r = 4$ antennas with spacing $d = \frac{\lambda}{2}$.

Fig. 34 and 35 show the transmitted signal and the received signals at the ULA.

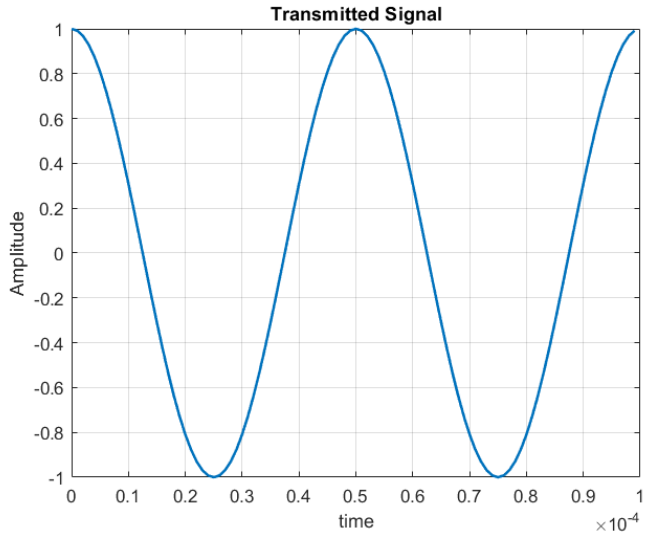

Figure 34 – Transmitted signal

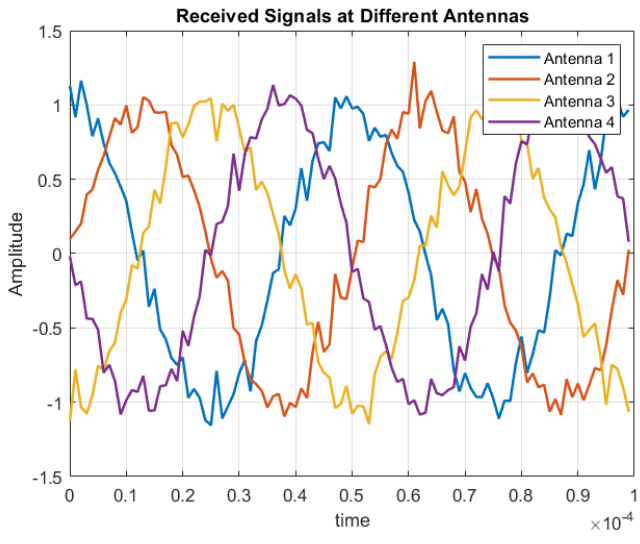

Figure 35 – Received signals at the ULA

Fig. 36 shows the beamformed signal after applying the weights. The amplitude of the signal is multiplied by 4 which is the number of antennas showing the beamforming effect.

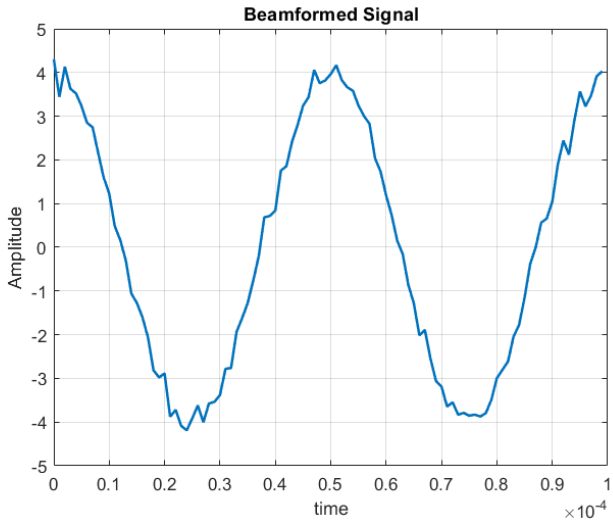

Figure 36 – Delay and sum beamformed signal

Fig. 37 and 38 show the results of performing the DOA at thetas from $-90°\ to\ 90°$.

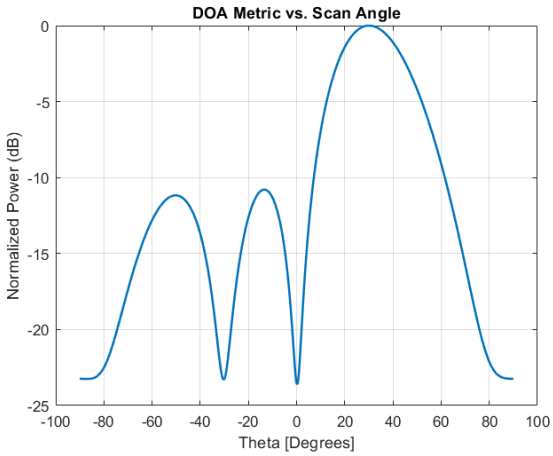

Figure 37 – Delay and sum DOA metric rectangular plot

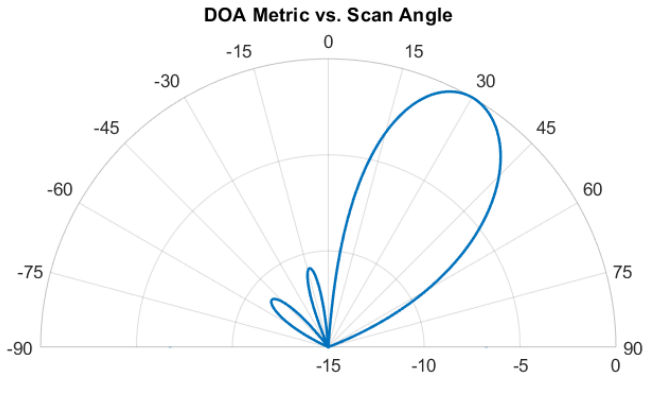

Figure 38 – Delay and sum DOA metric polar plot

For determining the direction of arrival, the variance (power) is calculated for the received signal at each scan angle and results are saved. The scan angle which corresponds to the highest variance is the angle of arrival. The weights are multiplied by the received signal at each scan angle then variance of the beamformed signal generated is calculated. After iterating at all scan angles the one with the highest variance is chosen.

```
theta_max = 30.0000
```

Figure 39 – MATLAB DOA calculation

3.2 MVDR

For a signal with AOA of $30°$ and a ULA with $N_r = 4$ antennas with spacing $d = \frac{\lambda}{2}$.

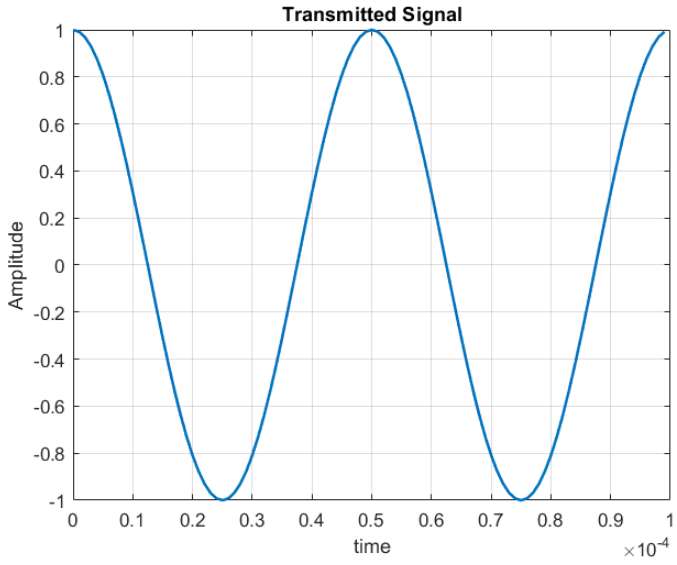

Figure 40 – Transmitted signal

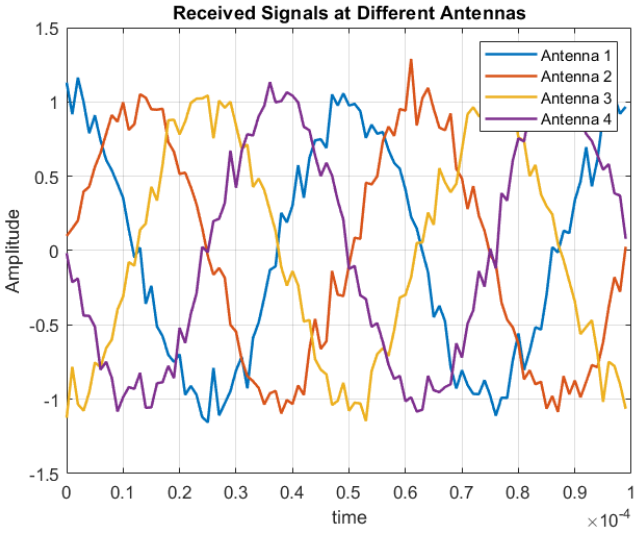

Figure 41 – Received signals at the ULA

Fig. 42 shows the beamformed signal after normalization.

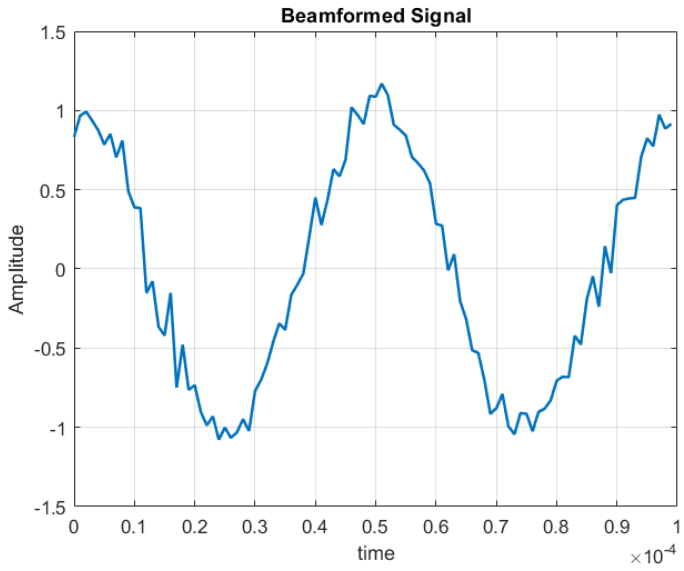

Figure 42 – MVDR beamformed signal

Fig. 43 and 44 show the results of performing the DOA at thetas from $-90°$ *to* $90°$.

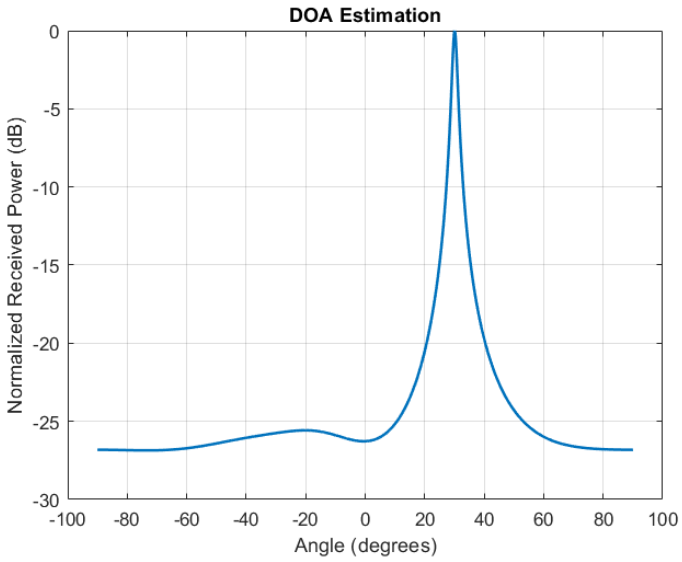

Figure 43 – MVDR DOA metric rectangular plot

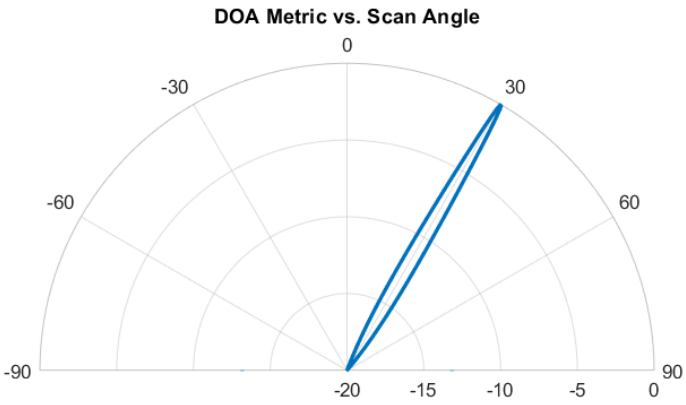

Figure 44 – MVDR DOA metric polar plot

Fig. 43 and 44 show that MVDR produces very sharp and directive beams.

For determining the direction of arrival, the variance (power) is calculated for the received signal at each scan angle and results are saved. The scan angle which corresponds to the highest variance is the angle of arrival. The weights are multiplied by the received signal at each scan angle then variance of the beamformed signal generated is calculated. After iterating at all scan angles the one with the highest variance is chosen.

```
theta_max = 30.0994
```

Figure 45 – MATLAB DOA calculation

3.3 Delay and sum Vs MVDR comparison

For comparing the performance of delay and sum vs MVDR 3 signals at three AOAs $\theta_1 = 20°, \theta_2 = 25°, \theta_3 = -40°$. The first two signals were chosen to arrive at two closely separated AOAs ($20° \; and \; 25°$) and the power of the third signal (arriving at $-40°$) is chosen to be 10 times less than the power of the first 2 signals.

The three signals are arriving at a ULA with $N_r = 8$ antennas with spacing $d = \frac{\lambda}{2}$.

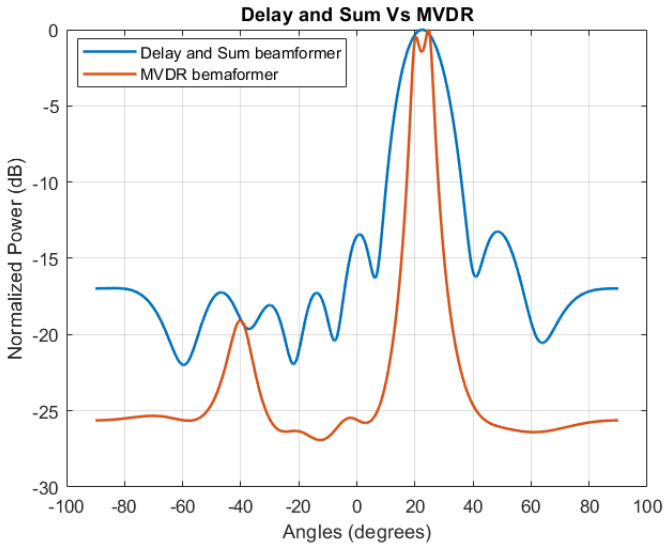

Figure 46 – Delay and sum Vs MVDR DOA metric rectangular plot

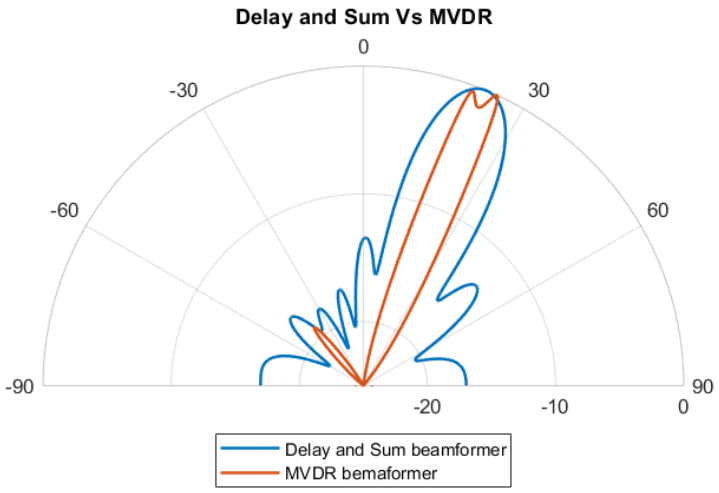

Figure 47 – Delay and sum Vs MVDR DOA metric polar plot

Fig. 46 and 47 show that the performance of MVDR is a lot better than the performance of the delay and sum because:

1- MVDR is more directive (the beam is more concentrated).
2- MVDR was able to detect and distinguish between the two signals arriving at the two close AOAs ($20°\ and\ 25°$) but delay and sum was not able to distinguish between them.
3- MVDR was able to detect the low power signal because its power level was higher than the side lobes level but delay and sum was not able to distinguish between the low power signal and its side lobes.

3.4 LCMV

To model the performance of LCMV multiple SOIs and interferers are simulated. Interferers are at AOAs of $-60°$, $-30°$, $0°\ and\ 45°$ and two SOIs are at $-45°\ and\ 30°$.

The signals are arriving at a ULA with $N_r = 8$ antennas with spacing $d = \frac{\lambda}{2}$.

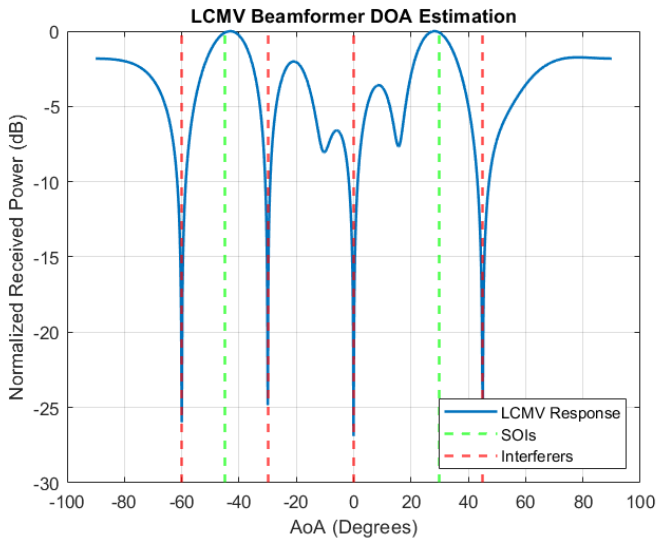

Figure 48 – LVMV DOA metric rectangular plot

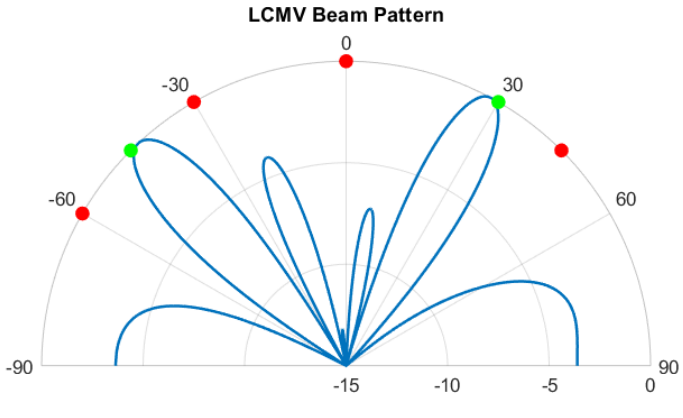

Figure 49 – LCMV DOA metric polar plot

Fig. 48 and 49 show that LCMV beamformer is a powerful tool that can be used to suppress interference and noise from multiple directions while simultaneously enhancing the signal of interest from multiple directions but the performance of the LCMV degrades near the edges of the antenna.

3.5 MUSIC Algorithm

Three SOIs coming at AOAs of $20°, 25° \; and \; -40°$ and the receiver is assumed to know the number of the signals arriving.

The signals are arriving at a ULA with $N_r = 4$ antennas with spacing $d = \frac{\lambda}{2}$.

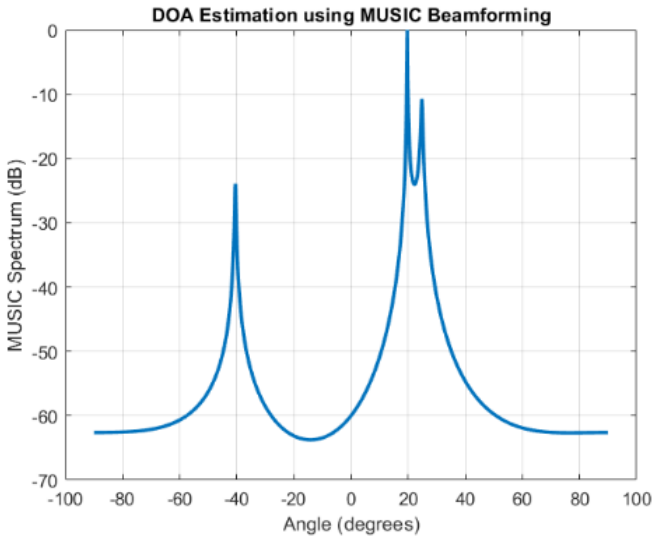

Figure 50 – MUSIC DOA metric rectangular plot

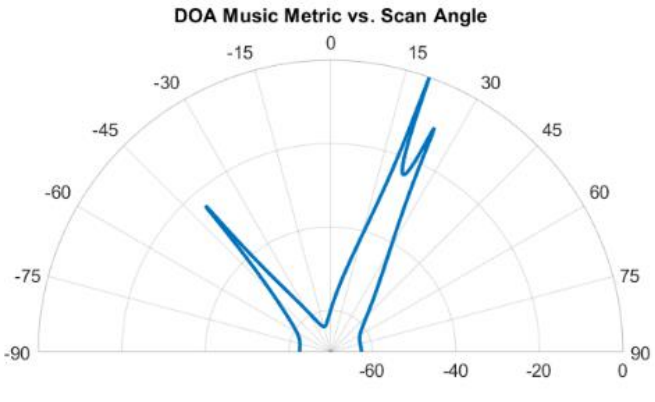

Figure 51 – MUSIC DOA metric polar plot

Fig. 50 and 51 show very precise results from MUSIC.

If the number of signals present is unknown, the eigenvalue magnitudes are sorted from highest to lowest. The eigenvalues associated with the noise-subspace are going to be the smallest, and they will all be around the same value, so we can treat these low values like a "noise floor", and any eigenvalue above the noise floor represents a signal.

Fig. 52 shows an estimation of the number of input signals for an 8-element array and 3 input signals. There are three points above the noise floor which represents the number of received signals. Underestimating this number will lead to missing signal(s) while overestimating will only slightly hurt performance as depicted in Fig. 53 and 54.

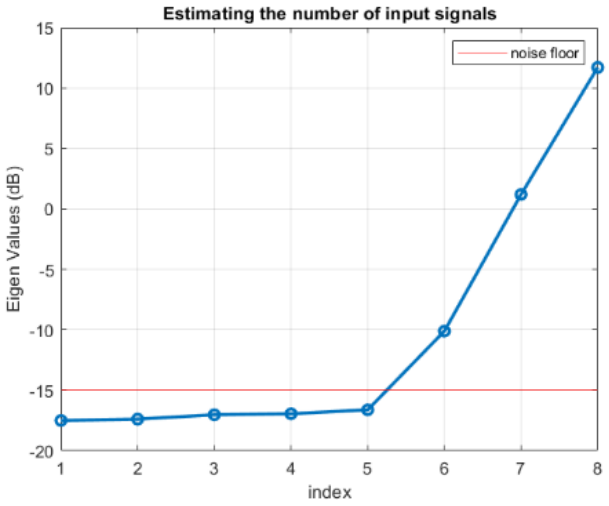

Figure 52 – MUSIC number of input signals estimation

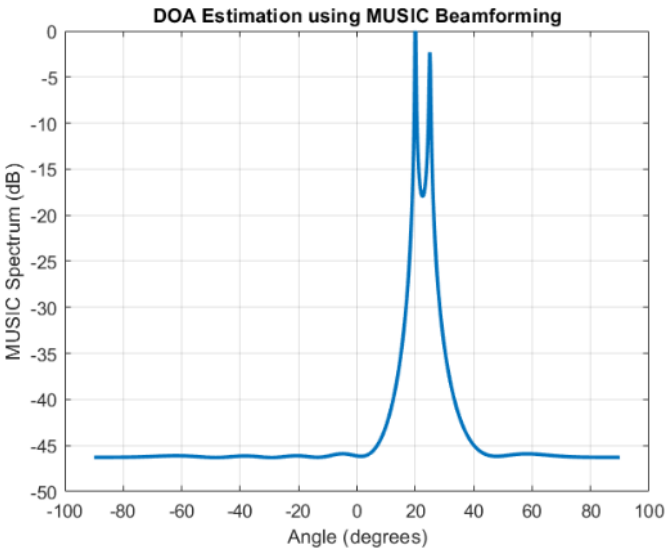

Figure 53 – Underestimating the number of input signals

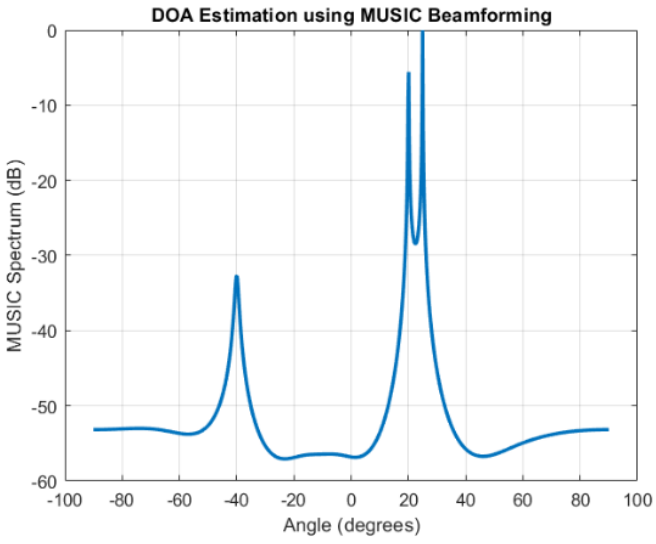

Figure 54 – Overestimating the number of input signals

An experiment worth trying with MUSIC is to see how close AOAs of two signals can be while still being able to distinguish between them, sub-space techniques are especially good at that. For three signals separated apart by 4 degrees ($AOAs\ at\ 20°, 24°\ and\ 28°$), Fig. 55 shows that we are able to distinguish between the different peaks despite being very close to each other.

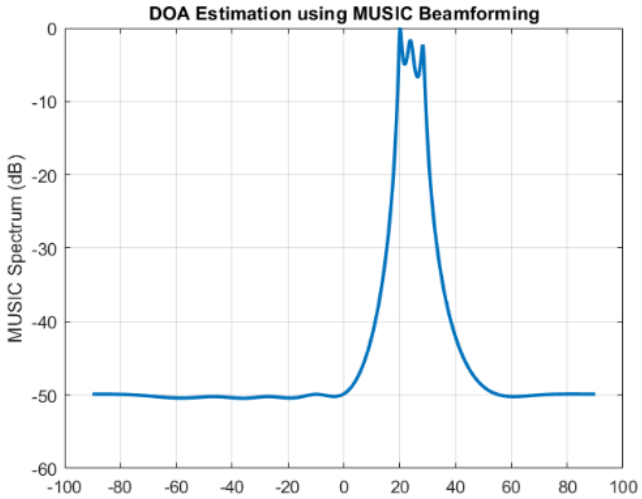

Figure 55 – MUSIC for closely separated signals

3.6 ESPRIT Algorithm

Three SOIs coming at AOAs of $20°, 25°\ and\ -40°$ the receiver is assumed to know the number of the signals arriving.

The signals are arriving at a ULA with $N_r = 4$ antennas with spacing $d = \frac{\lambda}{2}$.

To determine directions of arrival, it is necessary to:

- Achieve the decomposition in singular value of the data covariance matrix $R_{xx} = \frac{1}{N} XX^H$ where N is the number of observations.
- Estimate the dimension of the signal subspace Es in Eq. (40).
- Separate the eigenvectors corresponding to the signal subspace and form the matrices of sub-networks E_1 and E_2.
- Estimate the rotation operator Ψ in Eq. (45).
- Calculate the eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_k$ of the matrix Ψ in Eq. (46).
- Finally, calculate the AOAs in Eq. (47).

```
AoA = 20.56 degrees
AoA = 24.89 degrees
AoA = -40.11 degrees
```

Figure 56 – ESPRIT DOAs MATLAB calculation

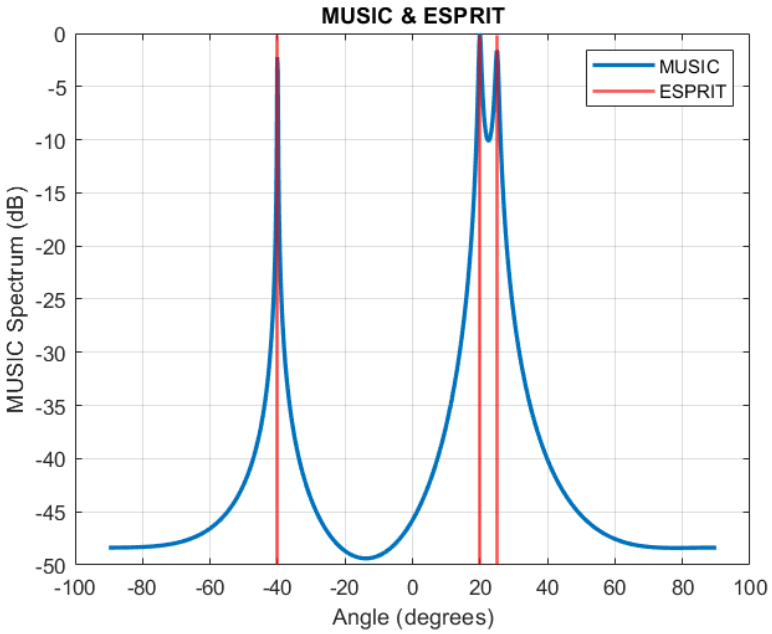

Figure 57 – MUSIC & ESPRIT DOA metric

Fig. 57 show that the performance of ESPRIT is very similar to the performance of MUSIC.

Ch 4: System Modeling

The first phase of this project involved a comprehensive literature review of MIMO and beamforming techniques, providing a solid foundation for understanding these concepts and identifying approaches relevant to system design. Moving forward, the focus will shift to the modeling and design of the system surrounding the beamforming block in both the transmitter (Tx) and receiver (Rx).

The next steps include developing models that represent the Tx and Rx systems, with particular attention to how the delay-and-sum beamforming technique can be implemented in hardware due to its simplicity and efficiency.

As a priority, the design and optimization of the Rx system will be undertaken first, ensuring its seamless integration with the beamforming block. This phase will involve analyzing hardware constraints, optimizing algorithms for real-time processing, and evaluating system performance in varying conditions.

4.1 System overview

To enable efficient implementation of DBF, a DBF architecture based on continuous-time band-pass $\Delta\Sigma$ modulators (CTBPDSMs) and bit-stream processing (BSP) is chosen.

In this architecture, IF signals are digitized by an array of CTBPDSMs to take advantage of direct IF sampling. By directly processing the un-decimated CTBPDSM digital outputs with BSP, digital down conversion (DDC) and phase shifting are implemented with only MUXs. Moreover, directly processing the CTBPDSM outputs avoids the need for multiple decimators for DBF. As a result, the architecture achieves low-power and area-efficient IF-sampling DBF. [25]

Delta-sigma modulation:

Delta-sigma (or sigma-delta) modulation is an **oversampling** method for encoding signals into low bit depth digital signals at a very high sample-frequency as part of the process of delta-sigma analog-to-digital converters (ADCs) and digital-to-analog converters (DACs). Delta-sigma modulation achieves high quality by utilizing a negative feedback loop during quantization to the lower bit depth that continuously corrects quantization errors and moves quantization noise to higher frequencies well above the original signal's bandwidth. Subsequent low-pass filtering for demodulation easily removes this high frequency noise and time averages to achieve high accuracy in amplitude.

A delta-sigma ADC encodes an analog signal using high-frequency delta-sigma modulation and then applies a digital filter to demodulate it to a high-bit digital output at a lower sampling-frequency. A delta-sigma DAC encodes a high-resolution digital input signal into a lower-resolution but higher sample frequency signal that may then be mapped to voltages and smoothed with an analog filter for demodulation. In both cases, the

temporary use of a low bit depth signal at a higher sampling frequency simplifies circuit design and takes advantage of the efficiency and high accuracy in time of digital electronics. [28]

DBF with Direct IF Sampling:

The concept of direct IF (or RF) sampling has arisen to enable digital-intensive receivers. By digitizing higher frequencies (i.e. IF or RF), most of the signal processing chain including down conversion and filtering is carried out in the digital domain. This enables perfectly matched digital I/Q down conversion as well as high-performance channel selection filtering. Furthermore, with direct IF sampling, the receiver is immune to flicker noise and DC offset.

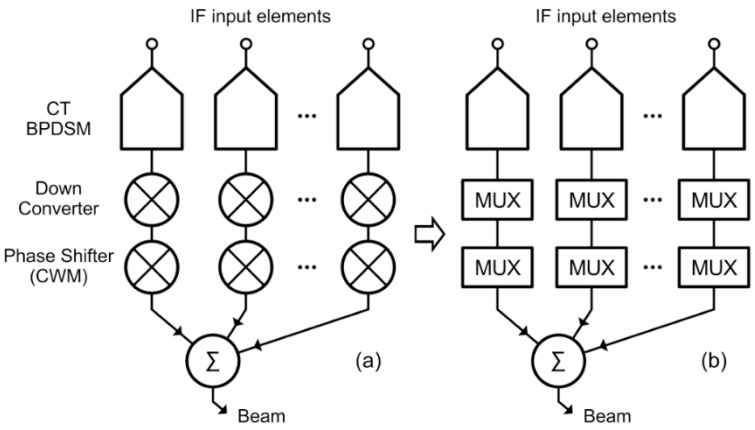

Figure 58 – (a) IF-sampling DBF and (b) its MUX-based implementation [25]

We implement IF-sampling DBF with an array of CTBPDSMs (sigma delta ADCs) as shown in Fig. 58a. IF input signals are directly digitized by CTBPDSMs, and digitally down converted to form baseband I/Q signals. The baseband I/Q signals are phase-shifted with CWM, and summed to create a beam. The IF-sampling DBF architecture normally requires several digital multipliers for DDC and CWM. However, thanks to the ΔΣ modulated low-resolution CTBPDSM digital outputs, the architecture is implemented very efficiently with MUXs as shown in Fig. 58b. As we will see next, multipliers are replaced with MUXs in BSP. As a result, both DDC and CWM are implemented with simple MUXs. [25]

Bit-Stream Processing DBF with ΔΣ Modulator Outputs:

In $\Delta\Sigma$ modulation, the combination of oversampling and noise shaping enables a high SNR modulator output with a **single-bit** (or low-resolution) quantizer. Conventionally, the low-resolution digital output of the $\Delta\Sigma$ modulator is low-pass filtered and decimated before further DSP (Fig. 59a). In the conventional approach, DSP is performed at a lower clock rate after decimation but at the cost of an increased word width. In BSP, on the other hand, the bit-stream modulator output is directly processed before decimation (Fig. 59b) to take advantage of the low word width. This approach introduces a multiplier-less digital filter with a single-bit $\Delta\Sigma$ modulator output.

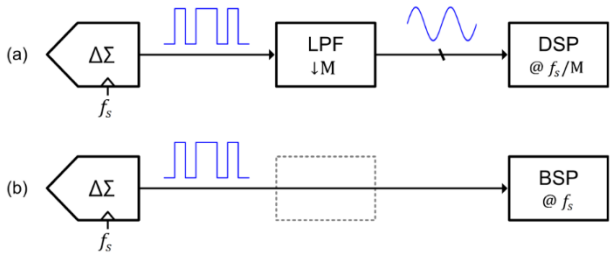

Figure 59 – (a) DSP after decimation (b) BSP [25]

A significant advantage of BSP is that it replaces bulky multipliers with simple MUXs. MUX-based multiplication with a bit-stream is described in Fig. 60. The bit-stream controls a 2:1 MUX to multiply the input bit-stream by a multi-bit coefficient, W, which is stored in a register. Depending on the value of the bit-stream, the 2:1 MUX output is selected to be either 0 or W. In this way, the 2:1 MUX output represents the result of multiplication of the bit-stream by W.

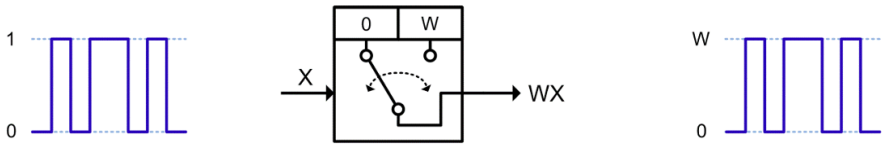

Figure 60 – Bit-stream multiplication with a 2:1 MUX

Another advantage of directly processing the CTBPDSM outputs in a multiple-input single-output system (e.g. beamformer) is that it reduces the number of decimators to just one. For multiple inputs and multiple ΔΣ modulators in conventional DSP (Fig. 61a), there is a decimator for each modulator. Because of this, the cost of decimation (by M) increases linearly with the number of inputs. In BSP, on the other hand, decimation is performed only once after all the digital signal paths are combined (Fig. 61b). Since decimation consumes a lot of power and requires a large area, the single decimation helps significantly reduce the power consumption and area of the entire system. [25]

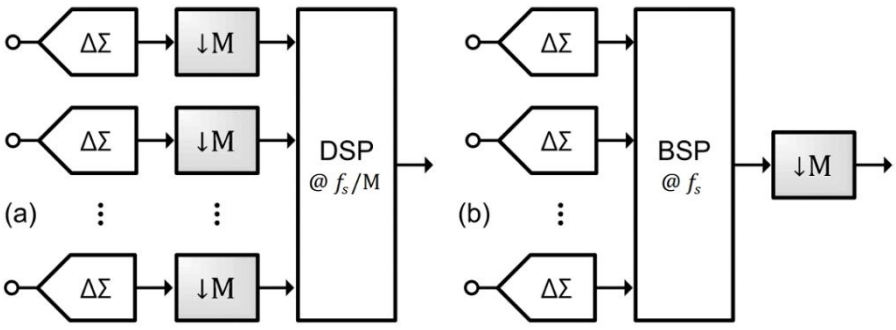

Figure 61 – (a) DSP with multiple decimators (b) BSP with a single decimator [25]

4.2 Receiver Model

After careful consideration of various architectural options for implementing a 5G Massive MIMO digital beamforming transceiver, a receiver model has been proposed that leverages the benefits of bit-stream processing (single-bit operations), intermediate frequency (IF) sampling, and Delta-Sigma modulation. This combination enables efficient, high-resolution signal processing while reducing power and hardware complexity.

Fig. 62 illustrates the proposed receiver architecture. The red box highlights the digital domain, which contains the core processing blocks to be designed and implemented. These digital blocks form the foundation of the beamforming and signal recovery stages and represent a critical component of the overall receiver system.

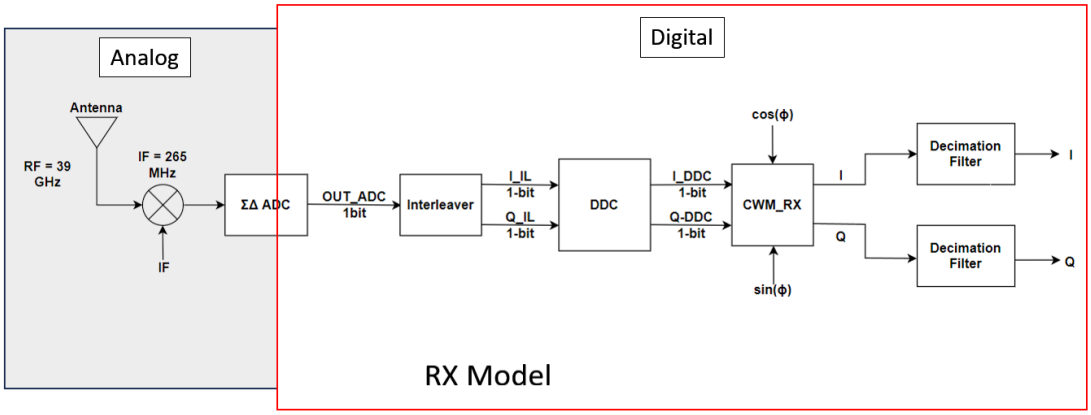

Figure 62 – Proposed RX model

To ensure compatibility with 5G standards and enable operation within Frequency Range 2 (FR2) — which spans from 24.25 GHz to 52.6 GHz — the baseband signal bandwidth is selected to be 50 MHz, aligning with typical bandwidth configurations used in 5G NR (New Radio) systems.

In accordance with 3GPP specifications, the corresponding sampling frequency is chosen to be 61.44 MHz. This value is derived based on standard numerologies, subcarrier spacing, and FFT sizes used in 5G systems. The 61.44 MHz sampling rate ensures proper alignment with channel bandwidth allocations, avoids aliasing, and allows for seamless interfacing with baseband signal processing chains. [29][30]

For operation using Delta-Sigma modulation, oversampling is essential to achieve noise shaping and maintain high signal fidelity. Therefore, an Oversampling Ratio (OSR) of 16 is selected. Accordingly, the decimation ratio in the receiver is also set to 16, allowing the system to reduce the sampling rate back to the Nyquist rate after digital filtering, while preserving the integrity of the baseband signal.

The following sections provide a detailed breakdown of each block within the digital domain of the proposed receiver. Each block is presented with the following structure:

- **Theoretical Background**: An overview of the underlying principles and signal processing concepts relevant to the block's function.

- **Proposed Design**: A description of the design approach adopted to meet the system requirements, including functional and architectural considerations.
- **Optimization**: Techniques applied to enhance performance, reduce complexity, or improve power and area efficiency.
- **MATLAB and Simulink Modeling**: Simulation and validation of the block's behavior using MATLAB and Simulink to ensure correct functionality prior to hardware implementation.
- **Final Design**: The finalized version of the block, prepared for integration into the overall system and ready for RTL implementation.

This structured approach ensures that each component of the receiver is rigorously analyzed, designed, and validated before moving forward in the development process.

4.2.1 Sigma Delta ADC

An efficient analog-to-digital converter ADC must be used due to its high consumption power and area as it works on very high frequencies. Especially when using multiple ADCs as in MIMO system that requires ADC for each antenna element, an optimized ADC is highly needed with small area and low power consumption. A 6th order Bandpass Sigma-Delta ADC provided by an external source (it's out of the digital domain) is employed to achieve high Signal-to-Noise Ratio (SNR) without increasing the sampling frequency fs or requiring a high oversampling rate. The advantage of Bandpass ADC over a Lowpass ADC is that the last requires doubling the number of ADCs compared to Bandpass ADCs to be used for I/Q branches, one for each Branch. The Over Sampling Ratio equals to 16, and the output of the ADC is a stream of one Bit (-1 or 1) as shown in Fig. 63.

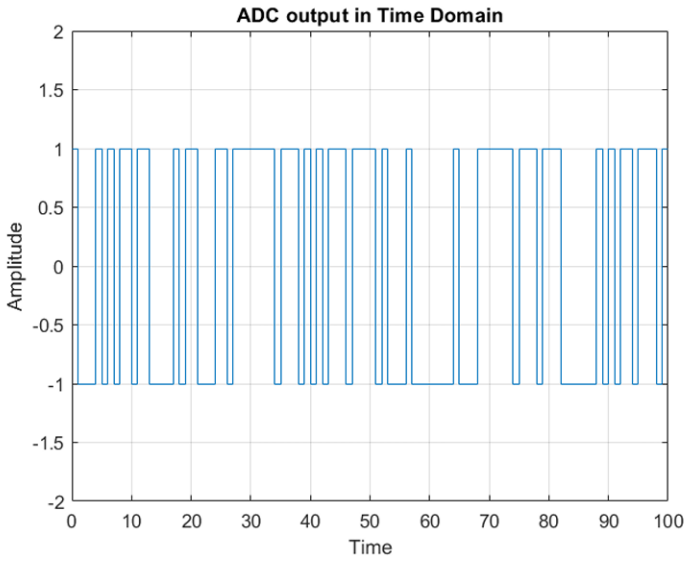

Figure 63 – ADC output in time domain

The bit stream output of the ADC can be effectively used in Bit-Stream Processing (BSP).

4.2.2 Interleaver

Interleaving is a critical technique in digital signal processing (DSP) systems, particularly for managing high data rates efficiently. Interleaving reduces the data rate output from the $\Sigma\Delta$ ADC, enabling more efficient downstream processing while maintaining system performance and accuracy.

Importance of Interleaving:

1- **Clock Rate Reduction** - Interleaving reduces the clock rate, halving it for example. This reduction lowers power consumption and decreases the complexity of digital circuitry.

2- **Power and Area Efficiency -** Interleaving significantly reduces power consumption and circuit area by enabling digital processing at lower speeds.

3- **Simplification of Digital Processing** - Interleaving simplifies Digital Down Conversion (DDC). Before using the Interleaver (IL), with carrier frequency $f_c = f_s/4$, the DDC values of sine and cosine waves were $(1, 0, 1)$, using a 3:1 MUX driven by the local oscillator (LO). By removing elements multiplied by zeros, the rate can be reduced with an Interleaver (a delay block and a down sampler by 2). This simplifies the DDC circuit implementation by using a 2:1 MUX driven by LO values -1 and 1.

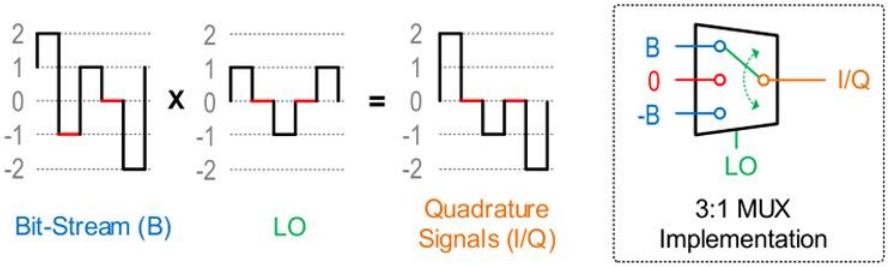

Figure 64 – DDC before adding IL [22]

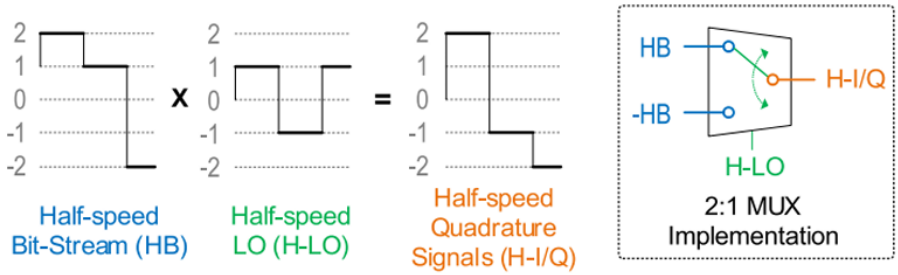

Figure 65 – DDC after adding IL [22]

Fig. 66 shows the setup of the interleaver, where odd-numbered data is output as I (In-phase) and even-numbered data as Q (Quadrature). This configuration enables the digital beamformer to operate at half speed without sacrificing performance. The interleaver is positioned at the beginning of the Digital Down Converter (DDC). So, it acts as a crucial component that reduces the clock rate and simplifies subsequent processing stages.

During implementation, ensuring proper generation of output bits involves synchronizing odd and even bits. Specifically, the even bits must be delayed appropriately to align with their corresponding odd bits for multiplication by the same local oscillator in the DDC. Without this synchronization, when odd bits enter the DDC, the even bits are not yet present, causing the oscillator to multiply by zero. This desynchronization leads to incorrect output. To address this, a delay block is inserted after the decimator in the even output branch as shown in Fig. 67. This ensures that even bits are correctly generated from their branch and synchronized in time with odd bits, thereby maintaining proper operation of the system.

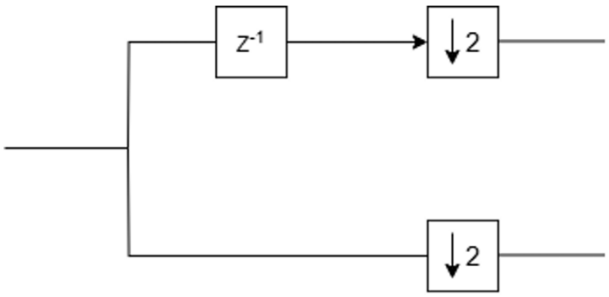

Figure 66 – IL block diagram initially

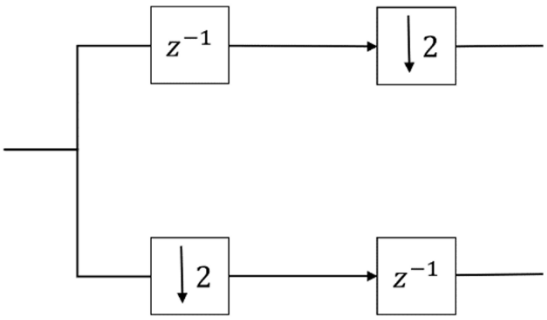

Figure 67 – Final IL block diagram

Interleaver design

Fig. 68 shows the simple design of the Interleaver implemented in Simulink.

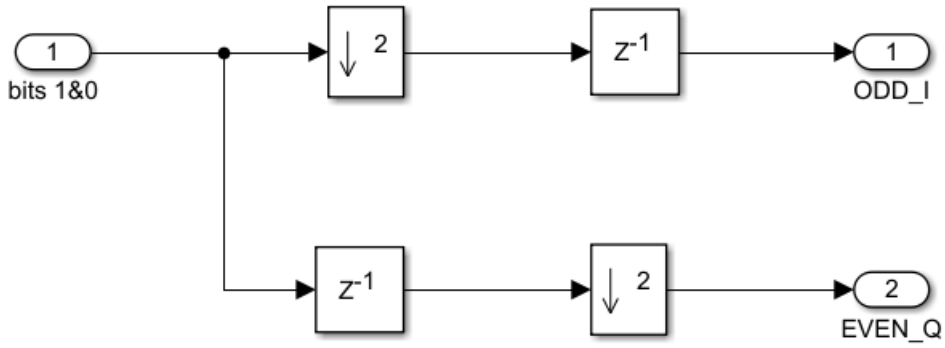

Figure 68 – Interleaver Simulink model

4.2.3 Digital Down Converter

The digital down converter is the same as analog down converter for quadrature amplitude modulated signals. Fig. 69 shows the block diagram; the constructing blocks are the same as the normal analog quadrature demodulator by multiplying the same received signal by sine and cosine waves with the same carrier frequency to move the signal back to baseband.

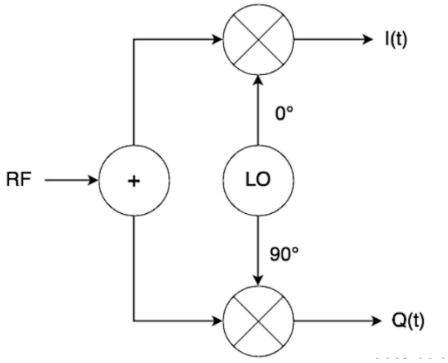

Figure 69 – DDC

An often-overlooked aspect in previous discussions is how sine and cosine waves with a specific carrier frequency are generated in the digital domain. Unlike the analog domain, where such signals are produced using local oscillators based on positive feedback principles, the digital approach is different. Typically, sine and cosine waves are stored in look-up tables (LUTs)—essentially ROMs preloaded with waveform values at a specific resolution. This resolution is usually determined by the desired accuracy of waveform sampling. However, in many digital signal processing applications—especially for up and down conversion—high-resolution waveforms are unnecessary. These applications prioritize the general behavior of the signal after multiplication with sine and cosine samples over precise waveform fidelity.

Because of this, only three key sample values are sufficient to represent the sine and cosine waves: $[-1, 0, 1]$. In the analysis, the carrier frequency (F_c) is selected to be one-fourth of the sampling frequency f_s, i.e., $f_c = f_s/4$. This choice leads to a simple and efficient sampling scheme where each full period of the sine wave is sampled four times, yielding $[0, 1, 0, -1]$, and the cosine wave is similarly sampled as $[1, 0, -1, 0]$. Sampling at this rate—every quarter cycle—not only avoids the need for complex LUTs but also significantly simplifies signal multiplication in the modulation and demodulation stages. With this method, the multiplication operation is greatly reduced in complexity. Instead of actual arithmetic, the operation becomes a simple selection (or multiplexing) task: multiplying by 1 passes the input sample unchanged, multiplying by -1 inverts the signal, and multiplying by 0 effectively nulls it. This dramatically reduces hardware complexity and processing time. Fig. 70 illustrates how the sine and cosine waveforms are generated digitally when $f_c = f_s/4$. [25]

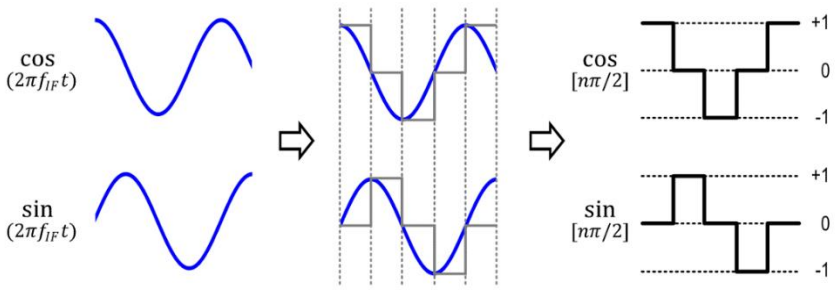

Figure 70 – Three-level I/Q LO sequences [25]

DDC design:

DDC is used to bring the signal back to the baseband from IF band by multiplying the signal by sine and cosine to generate the I (in phase) signal and the Q (Quadrature) signal respectively.

For $IF = f_s/4$, and due to the usage of the Interleaver and the benefit of removing the zeros, odd and even bit streams are multiplied by a square wave with frequency half of the sampling frequency $f_s/2$ as seen in Fig. 71.

Fig. 72 shows the final design of the DDC after optimization. In Fig. 72, multiplication of bitstreams by the square wave is done using XNOR gates, which replicate binary multiplication behavior in a 1-bit digital system.

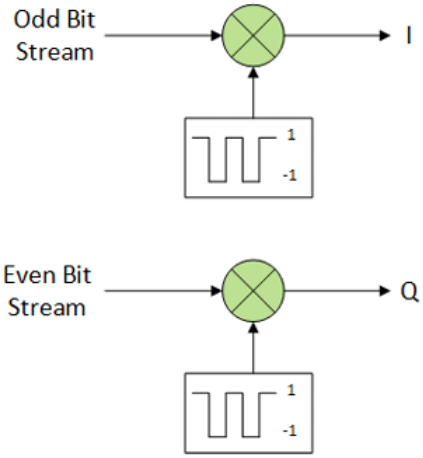

Figure 71 – DDC model

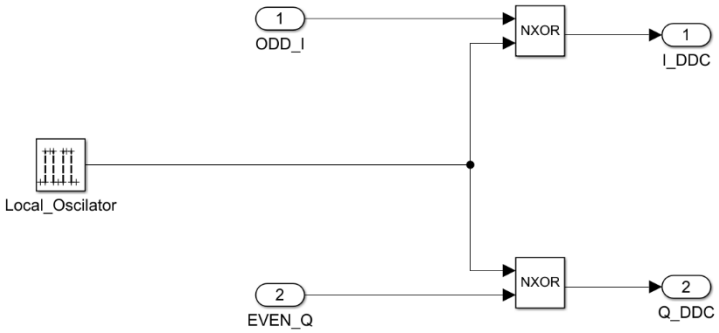

Figure 72 – DDC Simulink model

4.2.4 Complex weight Multiplication (CWM)

The Complex Weight Multiplication (CWM) block represents the simplest and most fundamental form of digital beamforming (delay & sum). It enhances signal strength in a desired direction by phase-aligning and summing signals received (or transmitted) across multiple antenna elements. This process effectively steers the beam toward a specific spatial direction, improving gain and signal quality.

In the receiver path, the CWM block operates on the In-phase (I) and Quadrature (Q) components of the signal, which are obtained after mixing with the local oscillator (LO) to convert the received RF signal to baseband. The complex baseband signal is then multiplied by a complex weight from a predefined weight matrix, where each weight corresponds to a specific antenna element and direction of interest.

This multiplication adjusts the phase (and optionally the amplitude) of each signal branch to constructively combine signals arriving from the desired direction and suppress interference or noise from others. The result is an enhanced I/Q signal pair, phase-aligned for optimal signal reception or transmission in the targeted direction.

By digitally controlling the phase of each down-converted signal x_k at the k-th element with DSP, element signals are constructively or destructively combined. To achieve a phase shift of θ, the baseband I/Q signals are scaled, and combined to generate phase-shifted I'/Q' outputs as follows:

$$I' = \cos(\theta)\, I + \sin(\theta)\, Q \tag{49}$$

$$Q' = -\sin(\theta)\, I + \cos(\theta)\, Q \tag{50}$$

When the I/Q signals are represented as a complex signal, the above operations are equivalent to multiplication by $e^{j\theta}$. For this reason, this technique is called complex weight multiplication (CWM).

For a uniformly spaced eight-element linear antenna array, a complex weight of $e^{jk\theta}$ adjusts the delay at the k-th element, and then all signal paths are combined to create a beam

$$= \sum_{k=0}^{7} x_k e^{jk\theta} \tag{51}$$

Since phase shifting with CWM is performed in the digital domain, DBF achieves the highest accuracy and flexibility. In addition, DSP algorithms can be easily applied in DBF for advanced functions including adaptive beamforming and array calibration.

Phase shifting with CWM is illustrated in Fig. 73. To achieve a phase shift of θ, baseband I/Q vectors are multiplied by weighting factors of combined to create phase-shifted I'/Q' vectors. [25]

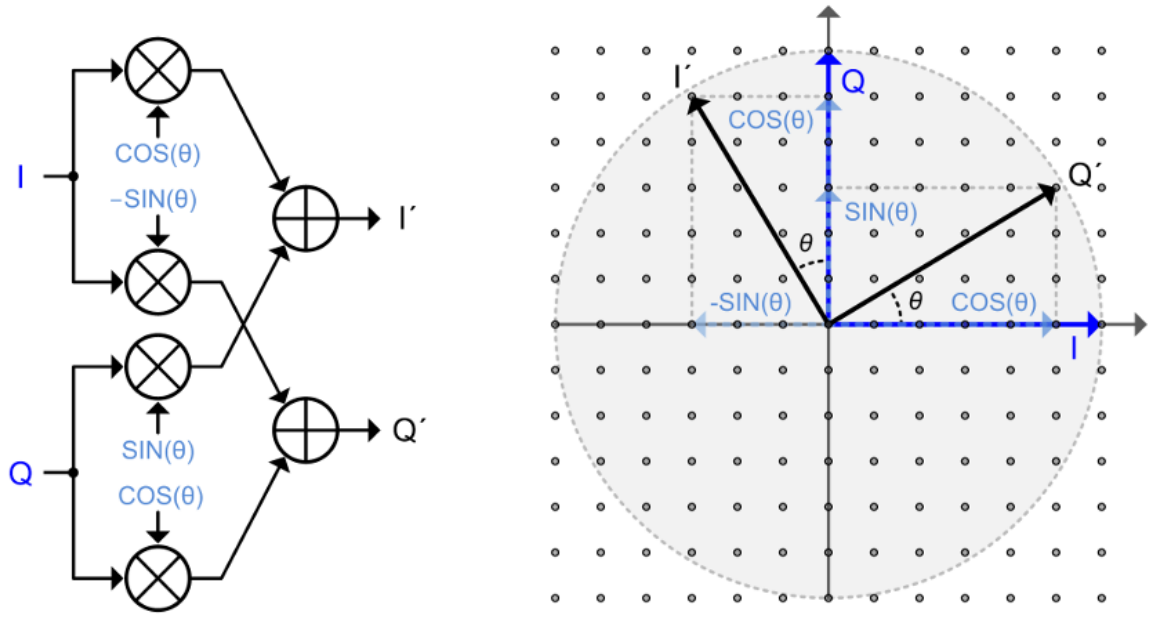

Figure 73 – Phase shifting with CWM [25]

CWM design

Fig. 74 presents the Simulink model of the Complex Weight Multiplication (CWM) block. As shown in the figure, the conventional multipliers are replaced by multiplexers (MUXs). This substitution is possible due to the single-bit nature of the input signal entering the CWM block, which allows for simplified hardware implementation. Instead of performing full multiplications, the system uses MUX-based logic to select precomputed weighted values, significantly reducing complexity and resource usage in the final RTL design.

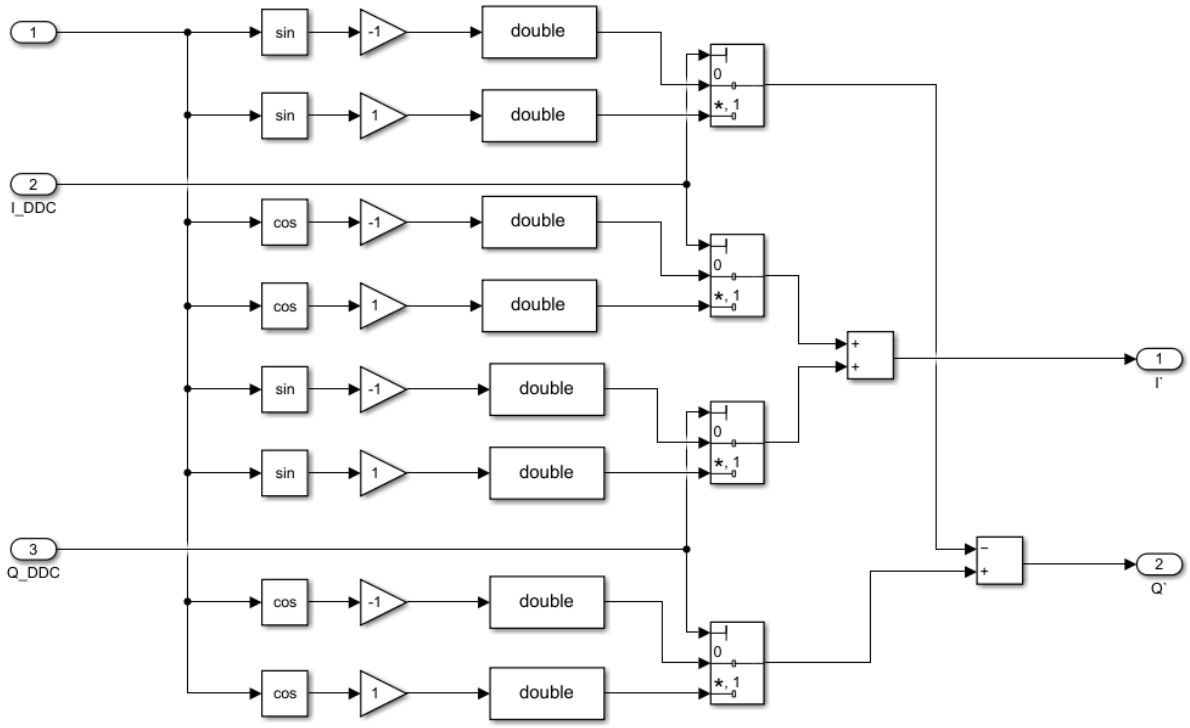

Figure 74 – CWM Simulink model

4.2.5 Decimation filter

Sampling rate conversion systems are used to change the sampling rate of a signal. The process of sampling rate decrease is called decimation, and the process of sampling rate increase is called interpolation. Two devices, the down-sampler and the up-sampler, are elements that change the sampling rate of the signal. The drawback of the down-sampling is the aliasing effect, whereas the up-sampling produces the unwanted spectra in the frequency band of interest. Decimation has to be performed in such a way as to avoid the effects of aliasing. [31]

Decimation requires that aliasing should be prevented. Hence, prior to down-sampling with the factor of M, the original signal has to be bandlimited to π/M. This means that the factor-of-M decimation has to be implemented in two steps:

(1) Band limiting of the original signal to π/M.

(2) Down-sampling by the factor-of-M.

Decimation of band limited signals

The decimation process is shown in Fig. 75a. The input signal is band limited with an FIR low-pass filter up to a highest frequency f_{max} and further decimated with a down sampler. The signal $z[nT_s]$ is sampled with a sampling period T_s or sampling frequency f_s. In the case of a decimation with factor M, only every M-th sample is retained, as shown in Fig. 75b.

As a result, the signal $y[nT_s{}']$ now has a sampling period $T_s{}' = MT_s$ and a lower sampling frequency equal to $f_s{}' = f_s M$. For the decimated signal to have the same information as the original useful signal, the decimation factor M must not exceed a certain value (To not violate the Nyquist condition).

$$f_s' \geq 2f_{max} \quad or \quad \frac{f_s}{M} \geq 2f_{max} \tag{52}$$

From this follows the condition for M

$$M \leq \langle \frac{f_s}{2f_{max}} \rangle \tag{53}$$

The ⟨ ⟩ denotes the smallest whole value of this fraction. As an example, for $f_{max} = 6\,kHz$ and $f_s = 100$ kHz, the ratio $\frac{fs}{2f_{max}}$ = 8.333 and thus $M = 8$ is selected. [2]

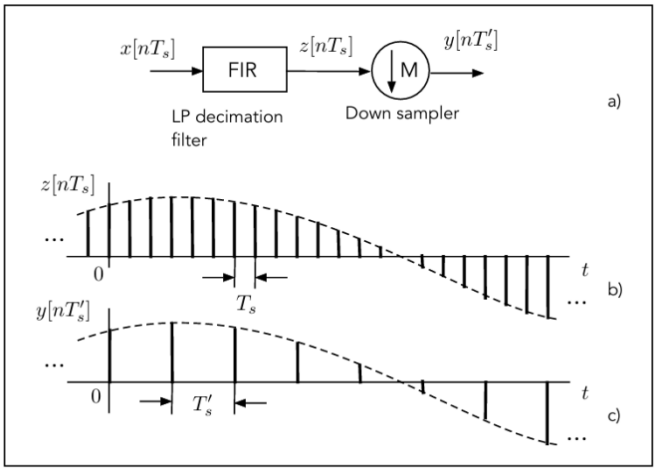

Figure 75 - The Decimation process [32]

Why the need of decimation filters?

Because of the use of the sigma-delta ADC which oversamples the signal above the Nyquist rate.

Oversampling sigma-delta analog-to-digital converters (ADC) have established themselves as a standard for high precision ADCs with the aid of digital filtering circuits. An oversampling ADC consists of a sigma-delta analog modulator and a digital decimation filter. The sigma-delta modulator modulates the analog input to an oversampled low resolution digital signal

at a frequency much higher than the Nyquist rate. The digital decimation filter digitally transforms the low-resolution oversampled signal into a high-resolution Nyquist-rate sampled signal. [33]

Proposed design

For this project the needed decimation factor is 16 and the decimation filter proposed consists of two stages:

1 - Comb filter with decimation factor of 8:

The comb filter acts as a low pass FIR filter that is used to band limit the signal before down sampling to ensure that Nyquist is not violated. The decimation filter cannot be implemented using only Comb Filter because its cut-off frequency is at very low frequency which can affect the bandwidth of the received signal.

2- Half band filter with decimation factor of 2.

The half band filter has a cut off frequency at $\pi/2$ which means large passband region so it's added after the comb filter to achieve the specification on the cut off frequency of the whole decimation filter.

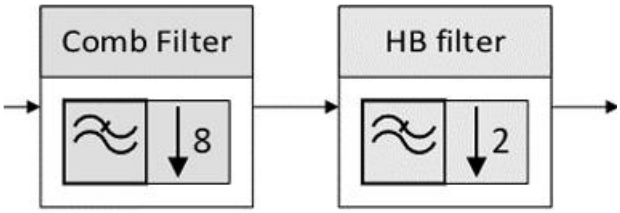

Figure 76 - The proposed Decimation Filter

Comb Filter

The simplest lowpass FIR filter is a boxcar or a moving-average (MA) filter whose impulse response is of a rectangle shape,

$$g_c[n] = \frac{1}{N} = \begin{cases} 1, & for\ 0 \le n \le N-1 \\ 0, & otherwise \end{cases} \quad (54)$$

where N is an integer. The z-domain representation results in the following transfer function $G_C(z)$,

$$G_c(z) = \frac{1}{N}\sum_{i=0}^{N-1} z^{-i} \quad (55)$$

Eq. (55) is recognized as the first N terms of the geometric series, whose closed form expression,

$$G_c(z) = \frac{1}{N}\frac{1-z^{-N}}{1-z^{-1}} \quad (56)$$

is fond suitable for an efficient implementation. According to Eq. (56), the filter $G(z)$ can be implemented by cascading the comb section $(1 - z^{-N})$ and the integrator section $1/(1 - z^{-1})$ thus leading to an extremely efficient device which performs the filtering task employing only two additions regardless of the filter length N.

A very poor magnitude characteristic of the comb filter is improved by cascading several identical comb filters. The transfer function $G_C^K(z)$ of the multistage comb filter composed of K identical single stage comb filters is given by

$$G_c^k(z) = \left[\frac{1}{N}\frac{1 - z^{-N}}{1 - z^{-1}}\right]^K \tag{57}$$

Fig. 77, 78 and 79 show the magnitude responses for the comb filter of the order N = 10, and K = 1, 2, 3, and 4, and the passband zoom.

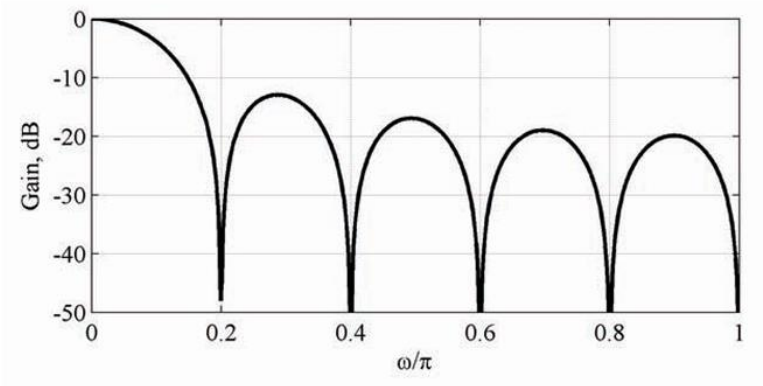

Figure 77 – Gain Response of the single comb filter for N = 10 [31]

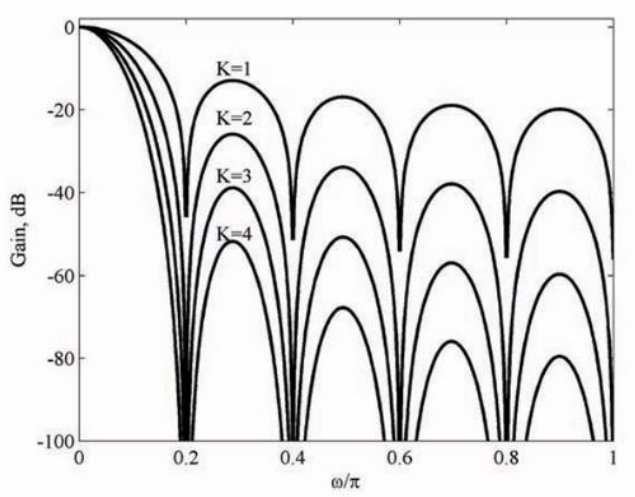

Figure 78 – Gain Response of the single comb filter for k = 1, 2, 3 and 4 [31]

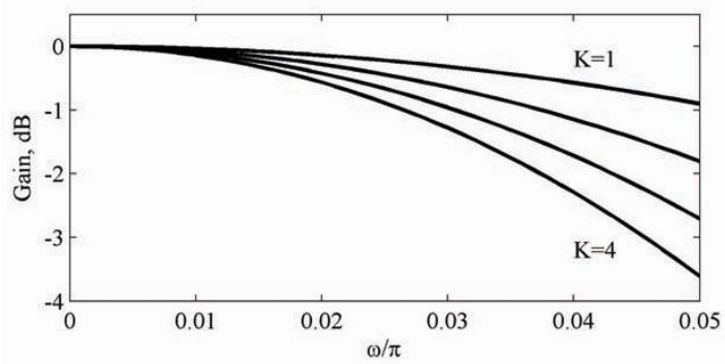

Figure 79 – Pass band response of the single comb filter for k = 1, 2, 3 and 4 [31]

Fig. 78 shows how the multistage realization improves the selectivity and the stop-band attenuation of the overall filter: the selectivity and the stopband attenuation are augmented

with the increase of the number of comb filter sections. The filter has multiple nulls with multiplicity equal to the number of the sections (K). Consequently, the stopband attenuation in the null intervals is very high. Fig. 79 illustrates a monotonic decrease of the magnitude response in the passband, called the passband droop. [31]

Comparisons of the Non-recursive and Recursive Algorithms:

Comb decimation filter has the following transfer function:

$$H(z) = \left(\frac{1 - z^{-N}}{1 - z^{-1}}\right)^k = \left(\sum_{i=0}^{N-1} z^{-i}\right)^k \tag{58}$$

where N is the decimation ratio and k indicate the order of comb filters (number of stages). The difference between the non-recursive and recursive algorithms is that they use different VLSI structures to implement the comb transfer function. So, the power consumption and operating speed of the circuit are different in these two algorithms.

Recursive Algorithm:

With the commutative rule the recursive algorithm with an IIR filter followed by a FIR filter is shown in Fig. 80. The switch in the figure indicates the reduction of the sampling rate by a factor of N. The IIR part is k integrators in cascade and works at oversampling frequency f_{os}. The FIR part is k differentiators in cascade and works at lower frequency f_{os}/N. The problems in this algorithm are the power consumption and speed limitation.

The problems in this algorithm are the power consumption and speed limitation. The output of a sigma-delta modulator usually has a short word length, say m bits, and a very high word rate, i.e. the oversampling rate. In order to avoid instability problems, a modulo arithmetic is used to determine the internal word length for most economic VLSI realization. According to the modulo arithmetic the minimum word length inside the IIR part is $m + k * \log_2 N$ bits. Since N is usually very large (32, 64, etc.) the IIR part has a long word length and has to operate at the very high oversampling frequency f_{os}. Therefore, large power consumption is caused by the IIR part. Meanwhile the IIR part limits the applicability of the recursive algorithm to very high frequency oversampling A/D converters. The observations are summarized in Fig. 81.

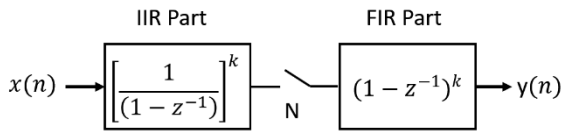

Figure 80 – Recursive algorithm for the comb filters [33]

	Input	Integrators	Output
Sampling rate	f_{os}	f_{os}	f_{os}/N
Word length	m	$m + k * log_2 N$	$m + k * log_2 N$

Figure 81 – Sampling rate and word length of the IIR filter in the recursive algorithm [33]

Non-recursive Algorithm:

Usually, the decimation ratio N is chosen to be M-th power of 2, i.e. $N = 2^M$. Then refer to Eq. (58), we have:

$$H(z) = \left(\sum_{i=0}^{N-1} z^{-i}\right)^k = \left(\sum_{i=0}^{2M-1} z^{-i}\right)^k = \prod_{i=0}^{M-1} \left(1 + z^{-2^i}\right)^k \tag{59}$$

By applying the commutative rule, the non-recursive algorithm is resulted, shown in Fig. 82. The switches in the figure indicate the reduction of sampling rates by a factor of 2. Every stage has the same low-order FIR filter but with a different sampling rate. The input $x(n)$ is from an oversampling sigma-delta modulator. Its word length W_d is assumed to be m bits. The word length increases through every stage by k bits but the word rate (sampling rate) decreases through every stage by a factor of 2 starting from the oversampling rate f_{os}. The observations are summarized in Fig. 83.

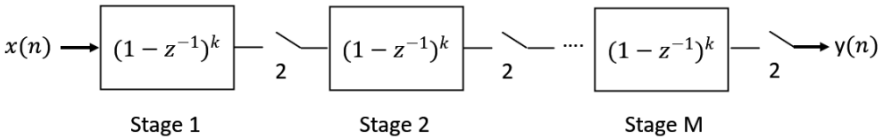

Figure 82 – Non-Recursive algorithm for the comb filters [33]

	Input	Output
Sampling rate	$f_{os}/2^{i-1}$	$f_{os}/2^i$
Word Length	$m + k * (i - 1)$	$m + k * i$

Figure 83 – Sampling rate and word length of the stage i in the non-recursive algorithm ($i = 1, 2, \ldots M$) [33]

From Fig. 83 it is seen that in the non-recursive algorithm, the word length is short when the sampling rate is high; and when the word length increases the sampling rate decreases. Reducing the sampling rates as early as possible helps to reduce the workload and thus the power consumption. From Fig. 81 and 83 it is clear that the frequency limitation is relaxed with the non-recursive algorithm. [33]

Comb filter design

For this project we will design a 4-stage comb filter ($k = 4$) with a decimation ratio $N = 8$ and $M = \log_2 N = \log_2 8 = 3$ so substituting in Eq. (59) we get,

$$H(z) = \prod_{i=0}^{2} \left(1 + z^{-2^i}\right)^4 = \left((1 + z^{-1}) \times (1 + z^{-2}) \times (1 + z^{-4})\right)^4 \tag{60}$$

Fig. 84 and 85 show the magnitude response for a single stage and a 4-stage comb filter.

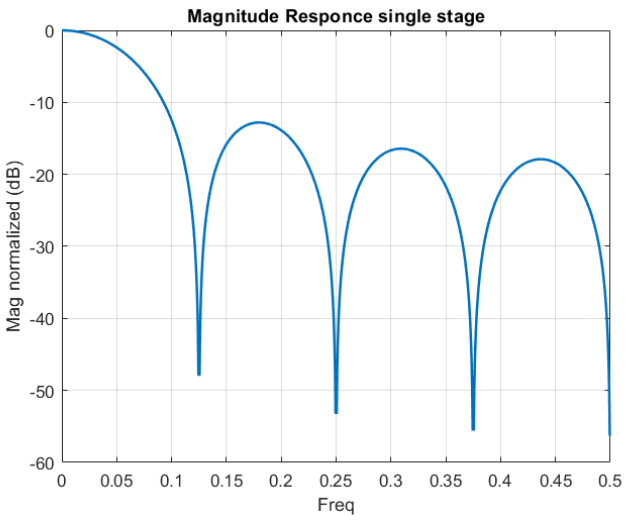

Figure 84 – The magnitude response of a single stage comb filter

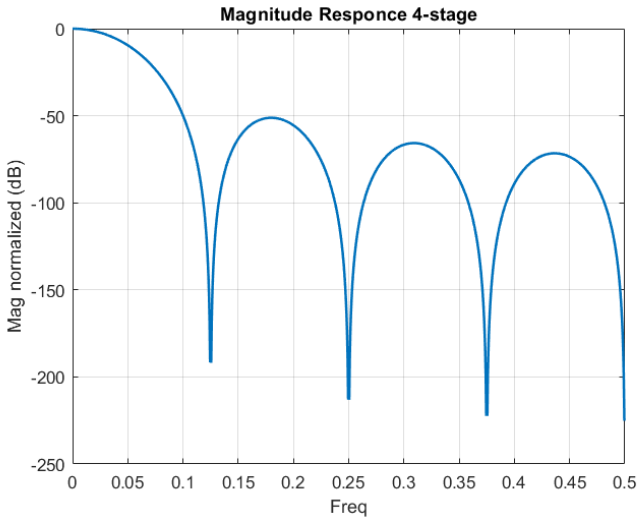

Figure 85 – The magnitude response of a 4-stage comb filter

For the implementation of the comb filter, we need to decompose Eq. (60) which can be decomposed to a summation of delayed elements of the input multiplied by different coefficients (FIR filter coefficients) which have this form,

$$H(z) = 1 + c_1 z^{-1} + c_2 z^{-2} + c_3 z^{-3} + c_4 z^{-4} + \cdots + c_n z^{-n} \quad (61)$$

Filter coefficients are obtained using MATLAB as follows,

H = 1×29

1 4 10 20 35 56 84 120 161 204 246 284 315 336 344 336 315 284 246 204 161 120 84 56 35 20 10 4 1

Filters frequency response can be written as follows,

$$H(z) = 1 + 4z^{-1} + 10z^{-2} + 20z^{-3} + \quad \ldots \quad + 20z^{-25} + 10z^{-26} + 4z^{-27} + z^{-28} \quad (62)$$

This equation can be implemented in Hardware by simply delaying the input and multiply it by the corresponding coefficient. The coefficients are symmetric ($c_1 = c_{28},\ c_2 = c_{27}$ …) which simplifies the hardware even more. MATLAB Simulink is used for hardware simulation. Fig. 86 and 87 show the design of the filter and its frequency response.

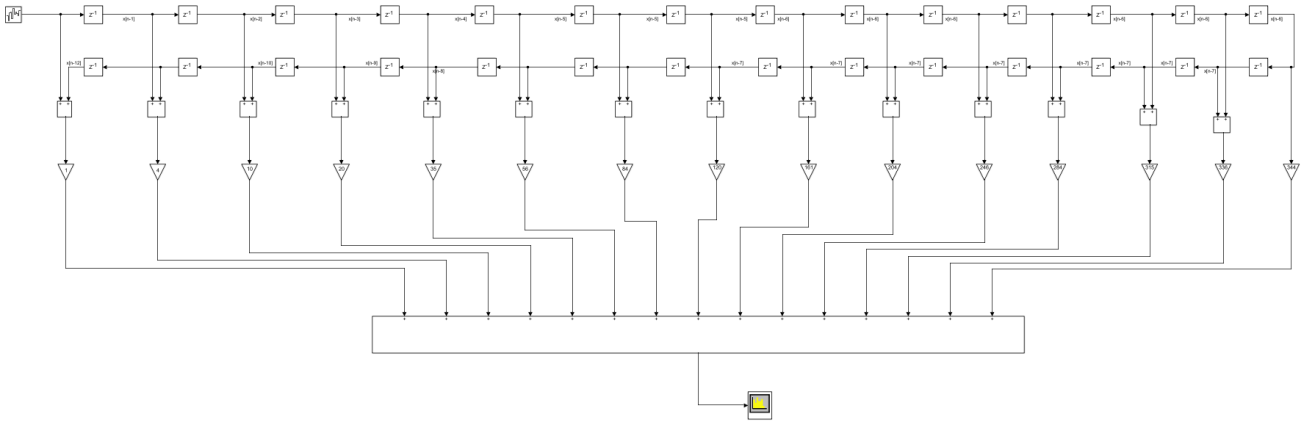

Figure 86 – Simulink model of the comb filter

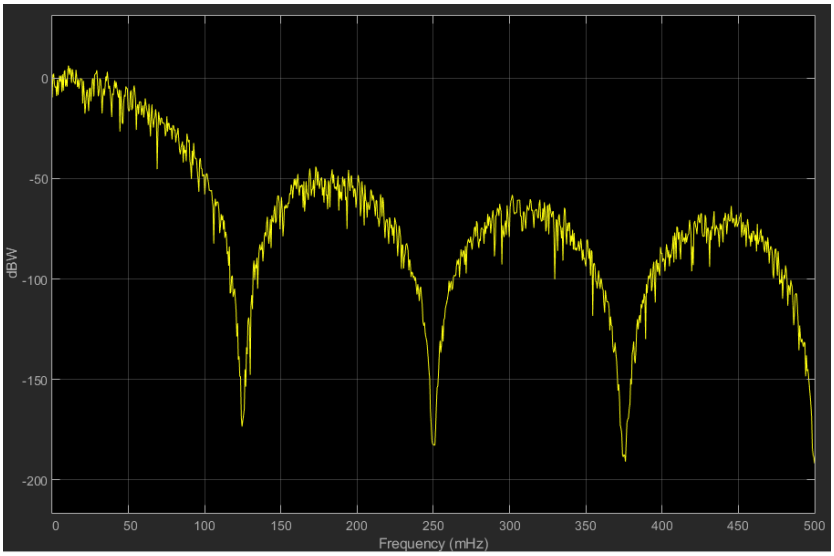

Figure 87 – Frequency response of the designed comb filter

Half Band Filter

A half band filter is an Lth-band filter with $L = 2$, and consequently, the half band filter divides the base band of the signal into two equal sub bands. In the linear-phase half band filter, half of the constants are zero-valued making the implementation very attractive.

Lth-band filter

Digital Lth-band FIR and IIR filters are the special classes of digital filters, which are of particular interest both in single-rate and multirate signal processing. The common characteristic of Lth-band lowpass filters is that the 6 dB (or 3 dB) cutoff angular frequency is located at π/L, and the transition band is approximately symmetric around this frequency.

In time domain, the impulse response of an Lth-band digital filter has zero valued samples at the multiples of L samples counted away from the central sample to the right and left

directions. Actually, an Lth-band filter has the zero crossings at the regular distance of L samples thus satisfying the so-called zero inter symbol interference property. Sometimes the Lth band filters are called the Nyquist filters.

The important benefit in applying Lth band FIR and IIR filters is the efficient implementation, particularly in the case $L = 2$ (half band) when every second coefficient in the transfer function is zero valued.

They are very popular in the sampling rate alteration systems, where they are used as decimation and interpolation filters in single-stage and multistage systems.

The filter transfer function $H(z)$ of the linear-phase Lth-band FIR filters can be expressed in the form

$$H(z) = \sum_{n=0}^{2K} h[n].z^{-n} \tag{63}$$

where, obviously, the filter length N is an odd number,

$$N = 2K + 1 \tag{64}$$

Since the filter is of a linear phase, the impulse response coefficients are symmetric,

$$h[2k - n] = h[n] \quad for\ n = 0, 1, \dots, 2k \tag{65}$$

The filter H(z) is an Lth-band filter if the impulse response coefficients satisfy the following conditions

$$h[k] = \frac{1}{L}, \quad h[k \pm rL] = 0 \quad for\ n = 1, 2, \dots, K/L \tag{66}$$

for the case $K = 10$ and $L = 4$. Here, the value of the central coefficient $h[10]$ is exactly $1/L = 1/4$, and the zero crossings occur at $n = 10 \pm 4$, and $n = 10 \pm 8$ as seen in 14a. Apparently, due to these conditions the frequency-domain restricts the passband of the lowpass filter to be smaller than π/L. The filter transition band is symmetric around the angular frequency $\omega_c = \pi/L$, and the zero phase frequency response has the value 0.5 at that frequency, see Fig. 88b.

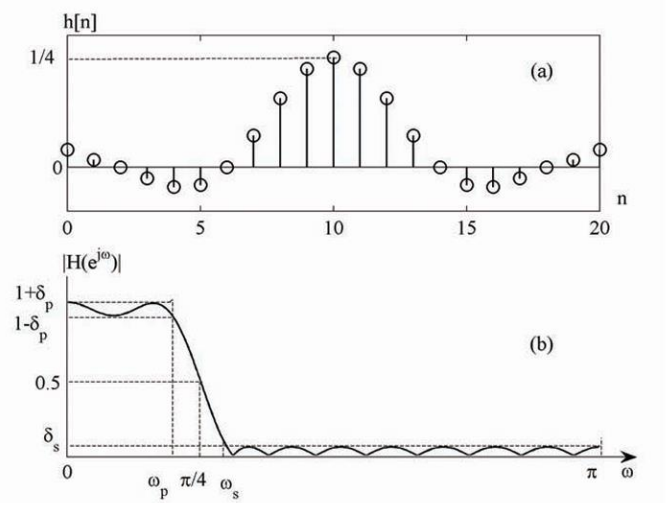

Figure 88 – 4th band filter with $k = 10$ [31]

Half band FIR filters are and Lth band filter with $L = 2$, hence substituting L by 2 and identifying the desired length K produces the half-band filter with cutoff frequency $\pi/2$.

Let K=5, N=11, Half-band filter can be presented as in Fig. 89.

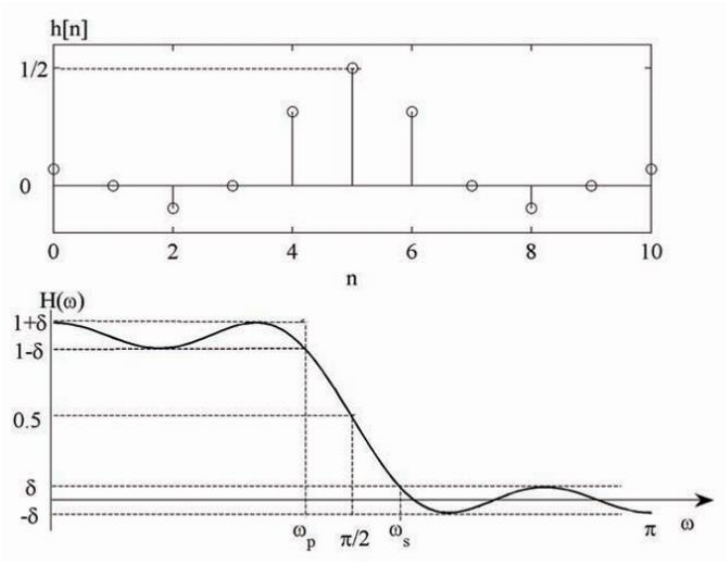

Figure 89 – Half band filter with $k = 5$ and $N = 11$ [31]

Half band Filter design

For achieving the specification on the cut off frequency of the decimation filter the length of the filter k is chosen to be 20 and the filter coefficients $h[n]$ are determined using Blackman window and obtained using MATLAB Filter Designer as follows,

$\boldsymbol{h[0]}$	$\boldsymbol{h[1]}$	$\boldsymbol{h[2]}$	$\boldsymbol{h[3]}$	$\boldsymbol{h[4]}$	$\boldsymbol{h[5]}$	$\boldsymbol{h[6]}$	$\boldsymbol{h[7]}$	$\boldsymbol{h[8]}$	$\boldsymbol{h[9]}$	$\boldsymbol{h[10]}$
0	0.0011	0	-0.0072	0	0.0263	0	-0.078	0	0.3078	0.5
$\boldsymbol{h[20]}$	$\boldsymbol{h[19]}$	$\boldsymbol{h[18]}$	$\boldsymbol{h[17]}$	$\boldsymbol{h[16]}$	$\boldsymbol{h[15]}$	$\boldsymbol{h[14]}$	$\boldsymbol{h[13]}$	$\boldsymbol{h[12]}$	$\boldsymbol{h[11]}$	
0	0.0011	0	-0.0072	0	0.0263	0	-0.078	0	0.3078	

Half the filter coefficients are zeros which reduce the number of multiplications. The coefficients are also symmetric which reduces this number even more.

Using these coefficients, the impulse response and the frequency response of the filter are obtained in Fig. 90 and 91.

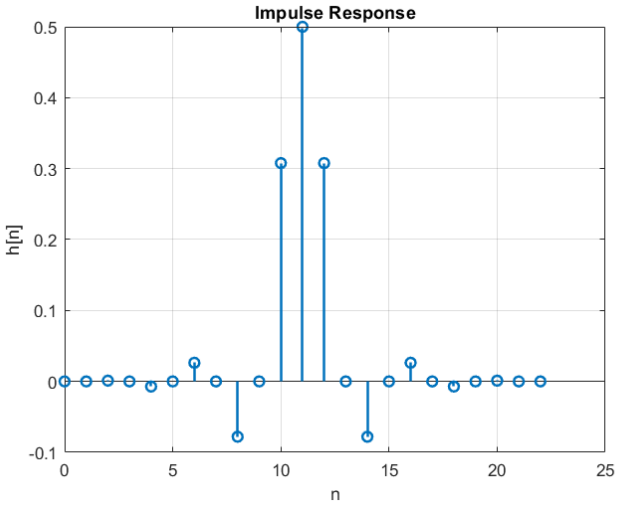

Figure 90 – Impulse response of the designed half band filter

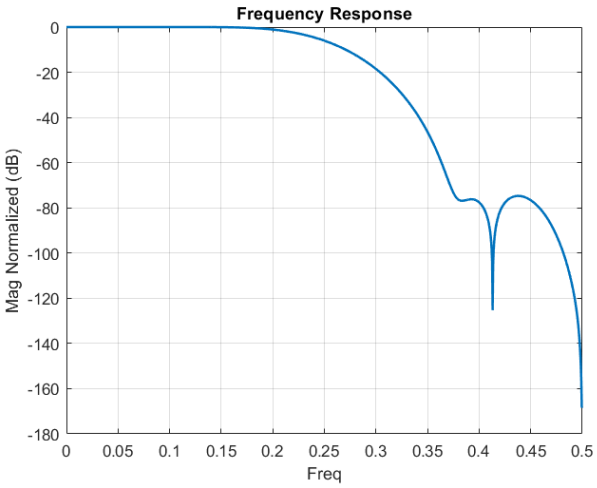

Figure 91 – Frequency response of the designed half band filter

The filters frequency response is,

$$H(z) = \sum_{n=0}^{2K} h[n].z^{-n} = h[0] + h[1]z^{-1} + h[2]z^{-2} + h[3]z^{-3} + \cdots + h[20]z^{-20} \quad (67)$$

but we have,

$$h[0] = h[2] = h[4] = h[6] = h[8] = h[12] = h[14] = h[16] = h[18] = h[20] = 0 \quad (68)$$

and we also have,

$$h[1] = h[19], \quad h[3] = h[17], \quad h[5] = h[15], \quad h[7] = h[13], \quad h[9] = h[11]$$

so,

$$H(z) = h[1](z^{-1} + z^{-19}) + h[3](z^{-3} + z^{-17}) + h[5](z^{-7} + z^{-13}) + h[7](z^{-7} + z^{-13}) + h[9](z^{-9} + z^{-11}) + h[10]z^{-10} \quad (69)$$

This equation can be implemented in Hardware by simply delaying the input and multiply it by the corresponding coefficient. MATLAB Simulink is used for hardware simulation. Fig. 92 and 93 show the design of the filter and its frequency response.

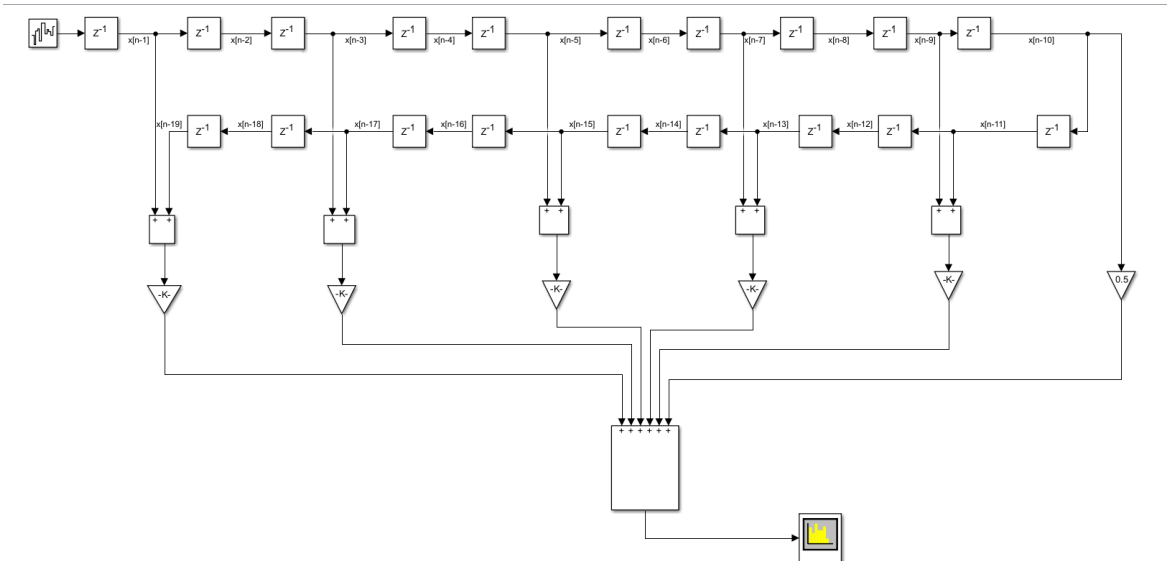

Figure 92 – SIMULINK model of the designed half band filter

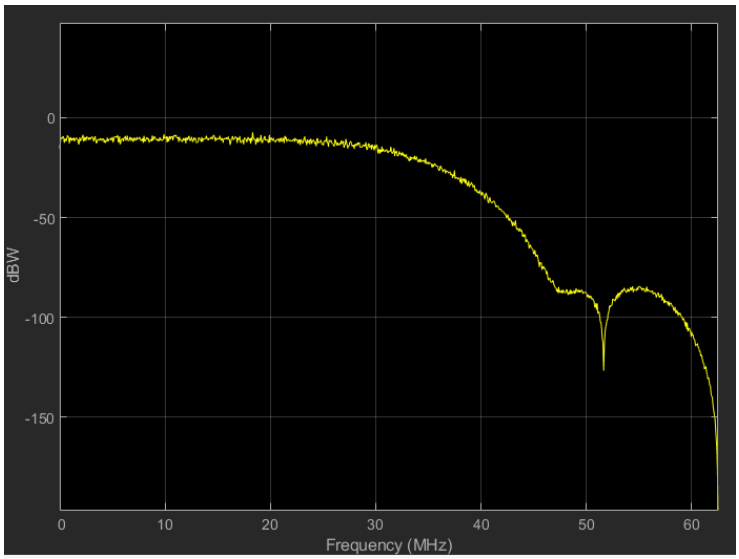

Figure 93 – Frequency response of the designed half band filter

Optimization

The hardware design of the comb and half band filters is not yet efficient for implementation due to the serial operation of the design which leads to a very high critical path delay, low throughput and high-power consumption. To mitigate these issues, Polyphase decomposition and Transposed FIR filter's structure are used.

Transposed FIR filters

Examining the Direct Form FIR filter:

Fig. 94 shows a five-tap FIR filter.

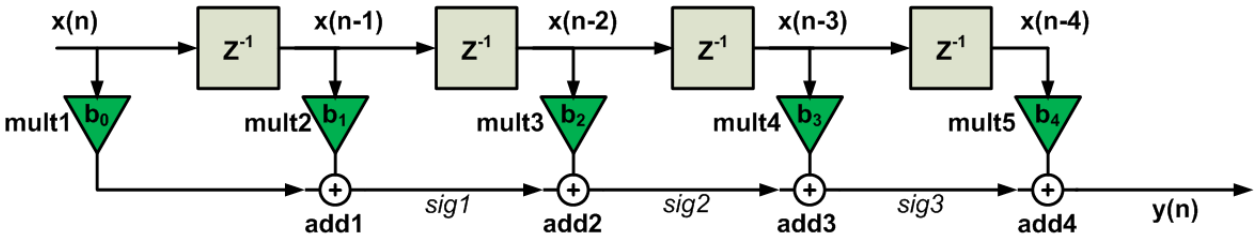

Figure 94 – Five-tap FIR filter [34]

Assume that multiplication and addition operations introduce a delay of T_{mult} and T_{add}, respectively. All the multipliers of Fig. 94 operate simultaneously, for example, when mult1 produces $x(n) \times b_0$, mult5 produces $x(n-4) \times b_4$. Hence, after applying a new input sample, the multipliers will output valid data only after a delay of T_{mult} (we assume that the delay of the registers is negligible). Then, we have to wait for add1 to produce a valid output, $sig_1 = x(n) \times b_0 + x(n-1) \times b_1$. This will introduce an extra delay of T_{add}. Similarly, each of the remaining additions will have a delay of T_{add} and the final output will be produced after a total delay of $T_{mult} + 4T_{add}$. Hence, the maximum sampling frequency of the filter can be estimated as

$$f_{sampling} = \frac{1}{T_{mult} + 4T_{add}} \tag{70}$$

Transposed realization of a FIR filter:

For a given system, we can achieve a new system structure by applying the "flow graph reversal" or the "transposition" theorem. The new structure is obtained by:

1- reversing the direction of all branches of the original system without changing the function of the branches.

2- reversing the roles of the input and output.

When applying the transposition method, it would be easier to consider the Signal Flow Graph (SFG) of the system. The SFG of Fig. 94 is shown in Fig. 95.

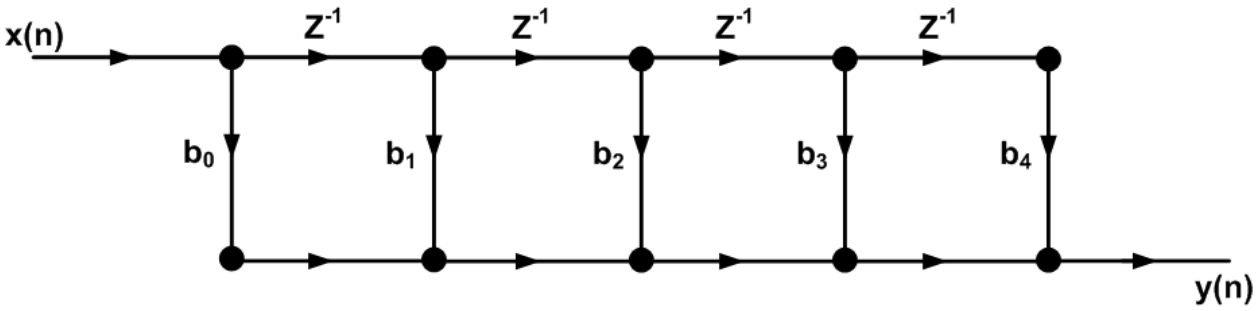

Figure 95 – SFG of a five-tab FIR filter [34]

Reversing the direction of the branches and the roles of the input and output, we obtain the new signal flow graph in Fig. 96.

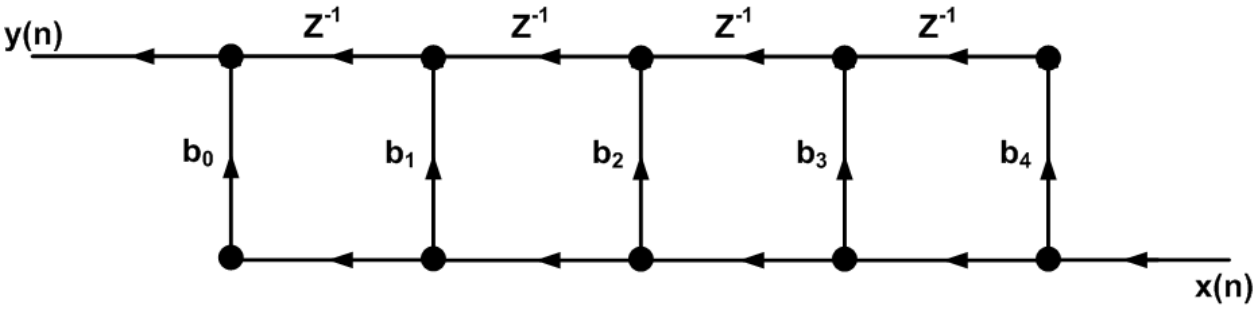

Figure 96 – SFG of a transposed five-tab FIR filter [34]

Since it is generally convenient to have the inputs on the left and the outputs on the right, we redraw the SFG of Fig. 96 as shown in Fig. 97.

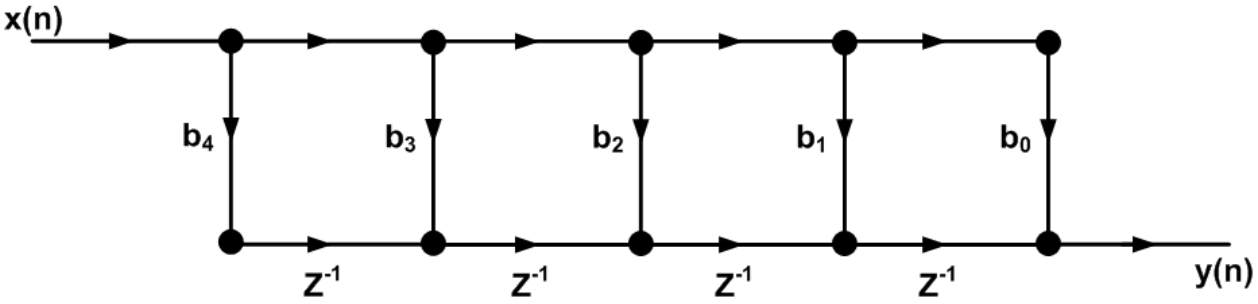

Figure 97 – Rearranged SFG of a transposed five-tab FIR filter [34]

The block diagram of the transposed-form FIR filter is shown in Fig. 98.

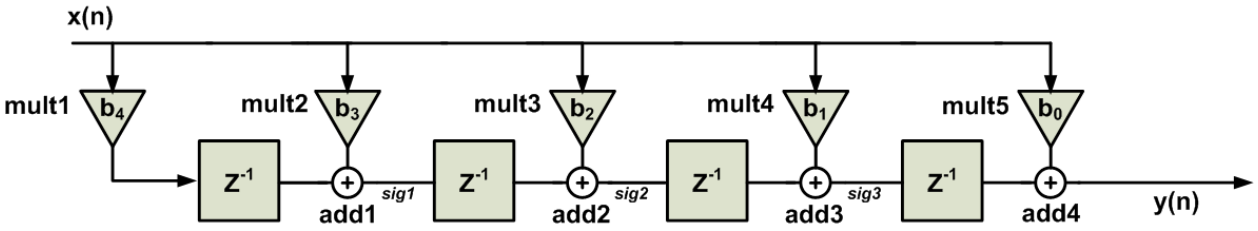

Figure 98 – The block diagram of the transposed-form FIR filter [4]

Comparing Fig. 98 with the direct form of Fig. 94, we observe that the order of the filter coefficients is reversed. Moreover, in the transposed form, the input reaches all of the multipliers at the same time. It is said that the input is broadcast to all of the multipliers. This is in contrast to the direct form structure where a given input sample reaches the multipliers at different clock cycles.

One of the most important features of the new structure is its **self-pipelined operation**. To understand this, let's see how a new sample is processed by the transposed structure. We'll examine the circuit in different clock cycles:

The 1st clock: Assume that, at the first clock edge, a new sample is applied to the filter. After a delay of T_{mult}, the multiplier mult1 will output $b_4 \times x(n)$.

The 2nd clock: At the second clock edge, the output of mult1 will be stored in the leftmost register. The register introduces a unit delay; hence, the content of the register will be $b_4 \times x(n-1)$. This means that the register stores b_4 multiplied by the previous sample of the input. To further clarify, note that, we are at the second clock cycle and the stored value corresponds to the sample taken in the first clock cycle.

Besides, with a delay of T_{mult}, mult2 will output $b_3 \times x(n)$. Hence, with a delay of $T_{mult} + T_{add}$ after the second clock edge, we have $sig_1 = b_4 \times x(n-1) + b_3 \times x(n)$.

The 3rd clock: at the third clock edge, sig_1 will be stored in the corresponding register. Moreover, mult3 will output $b_2 \times x(n)$. Hence, with a delay of $T_{mult} + T_{add}$ after the third clock edge, we have

$sig_2 = b_4 \times x(n-2) + b_3 \times x(n-1) + b_2 \times x(n)$.

The 4th clock: with a delay of $T_{mult} + T_{add}$ after the fourth clock edge, we have

$sig_3 = b_4 \times x(n-3) + b_3 \times x(n-2) + b_2 \times x(n-1) + b_1 \times x(n)$.

The 5th clock: with a delay of $T_{mult} + T_{add}$ after the fourth clock edge, we have

$y(n) = b_4 \times x(n-4) + b_3 \times x(n-3) + b_2 \times x(n-2) + b_1 \times x(n-1) + b_0 \times x(n)$.

This is the value of the output during the 5th clock cycle.

If we consider the transposed structure during different clock cycles, we observe that the registers are storing the final result calculated by all the previous stages. Hence, these previous stages can be used to process new samples. For example, during the second clock cycle discussed above, the leftmost register stores the output of the mult1 multiplier. Hence, mult1 is, in fact, freed to work on the input sample corresponding to the second clock cycle. The result of this calculation will go through the pipeline shown in Fig. 99 and, after the steps discussed above, the corresponding output will be produced. The transposed structure is inherently a **pipelined** implementation. [34]

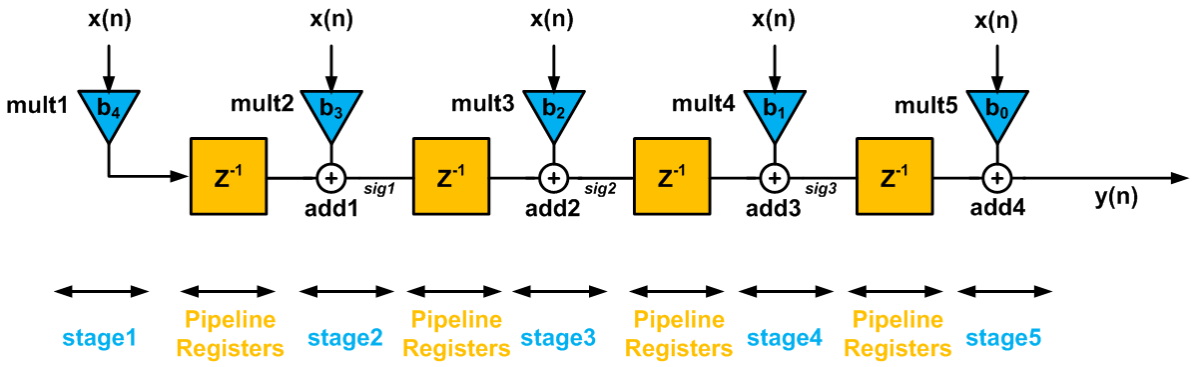

Figure 99 – The pipelined transposed-form FIR filter [34]

Transposed structure in this project:

The inherent pipelining in the transposed structure offers great improvement in performance so it is used in the implementation of the comb and half band filters.

Fig. 100 and 101 show the Simulink model of the transposed implementation of the comb filter and its frequency response respectively. The response of the transposed implementation of the filter in Fig. 101 matches the response of the direct form implementation of the filter in Fig. 87.

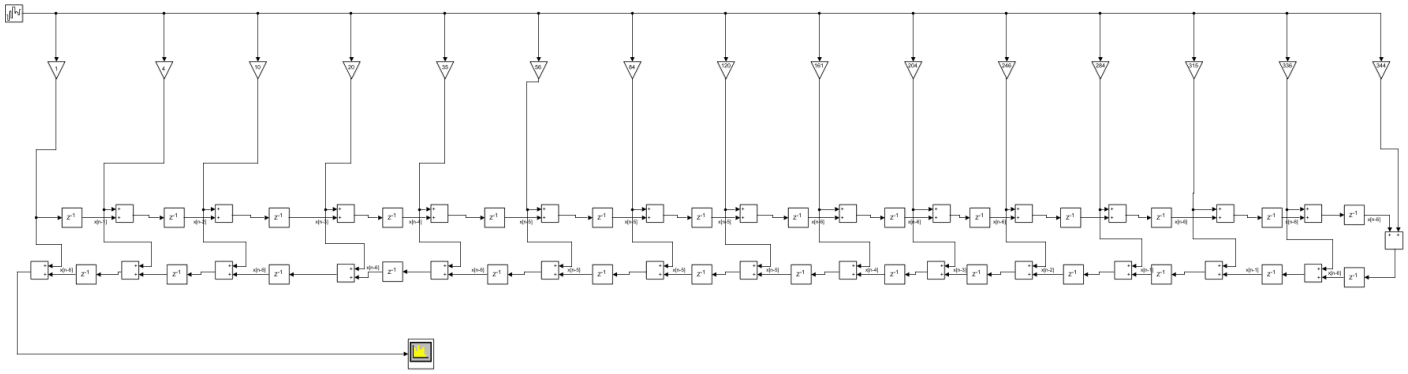

Figure 100 – Simulink model of the transposed implementation of the comb filter

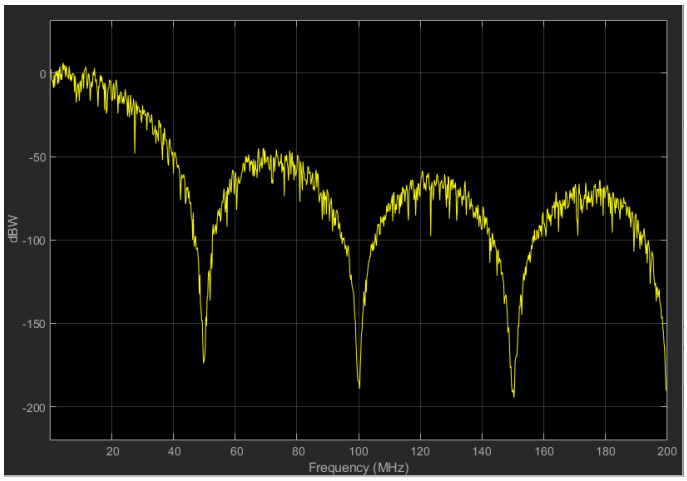

Figure 101 – Frequency response of the transposed implementation of the comb filter

Fig. 102 and 103 show the Simulink model of the transposed implementation of the half band filter and its frequency response respectively. The response of the transposed implementation of the filter in Fig. 103 matches the response of the direct form implementation of the filter in Fig. 93.

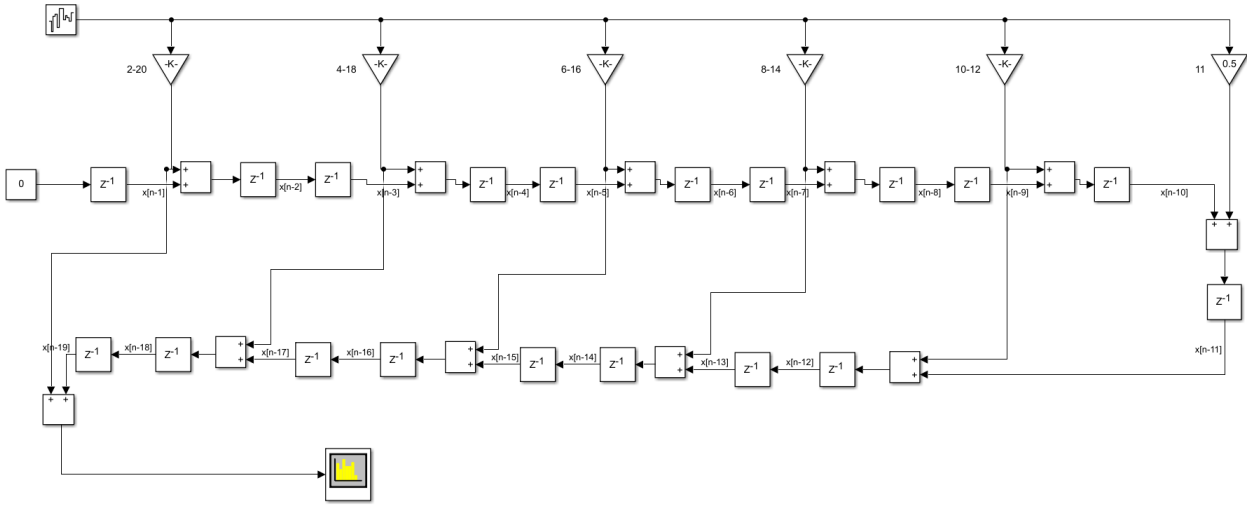

Figure 102 – Simulink model of the transposed implementation of the half band filter

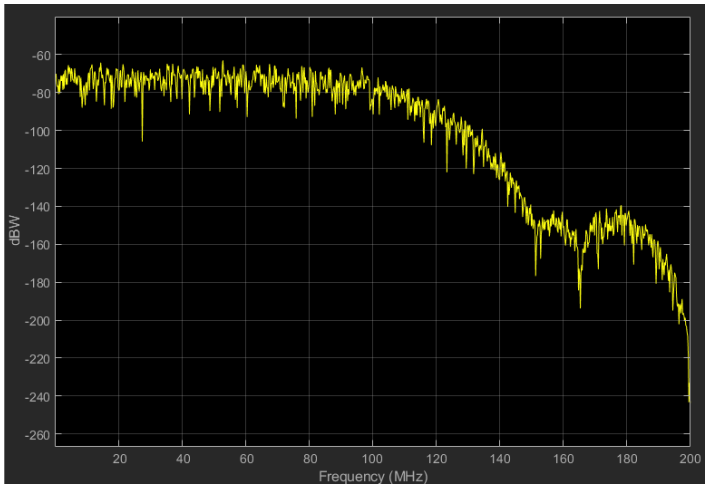

Figure 103 – Frequency response of the transposed implementation of the half band filter

Polyphase decomposition

The polyphase filters allow first down sampling and then implementing the filters at the lower sampling frequency.

The polyphase realization of FIR filters:

The polyphase realization of the FIR filter is a decomposition, which leads to a conversion into a parallel structure. To illustrate this approach a causal FIR filter of order 8 with nine coefficients $H(z)$ is first coverted to a structure with two parallel filters.

$$H(z) = h[0] + h[1]z^{-1} + h[2]z^{-2} + h[3]z^{-3} + h[4]z^{-4} + h[5]z^{-5} + h[6]z^{-6} + h[7]z^{-7} + h[8]z^{-8} \quad (71)$$

As a sum of two filters, with one containing the **even**-indexed coefficients and the other containing the **odd**-indexed coefficients the following decomposition results:

$$H(z) = (h[0] + h[2]z^{-2} + h[4]z^{-4} + h[6]z^{-6} + h[8]z^{-8}) + (h[1]z^{-1} + h[3]z^{-3} + h[5]z^{-5} + h[7]z^{-7}) \quad (72)$$

$$= (h[0] + h[2]z^{-2} + h[4]z^{-4} + h[6]z^{-6} + h[8]z^{-8}) + z^{-1}(h[1] + h[3]z^{-2} + h[5]z^{-4} + h[7]z^{-6}) \quad (73)$$

With the notations

$$E_0(z) = h[0] + h[2]z^{-1} + h[4]z^{-2} + h[6]z^{-3} + h[8]z^{-4}$$
$$E_1(z) = h[1] + h[3]z^{-1} + h[5]z^{-2} + h[7]z^{-3} \quad (74)$$

Eq. (20) now becomes:

$$H(z) = E_0(z^2) + z^{-1}E_1(z^2) \quad (75)$$

In a similar manner, by grouping the terms of Eq. (71) differently, you can get a different form:

$$H(z) = E_0(z^3) + z^{-1}E_1(z^3) + z^{-2}E_2(z^3) \quad (76)$$

Where now:

$$E_0(z) = h[0] + h[3]z^{-1} + h[6]z^{-2}$$
$$E_1(z) = h[1] + h[4]z^{-1} + h[7]z^{-2}$$
$$E_2(z) = h[2] + h[5]z^{-1} + h[8]z^{-2} \quad (77)$$

In the general case, an M-branch polyphase decomposition of an FIR filter of order $N-1$ with the transfer function

$$H(z) = \sum_{n=0}^{N-1} h[n]z^{-n} \quad (78)$$

result in

$$H(z) = \sum_{m=0}^{M-1} z^{-m}\, E_m[z^m] \quad (79)$$

$$E_m(z) = \sum_{n=0}^{\lceil N/M \rceil} h[Mn+m]z^{-n}, \qquad 0 \le m \le M-1 \quad (80)$$

Here $\lfloor N/M \rfloor$ is the integer part of N/M and $h[n] = 0$ for $n \geq N$. In Fig. 104 three different structures for a polyphase decomposition with M = 2, 3 and 4 branches are shown.

In Fig. 105 it is shown how the coefficients of an FIR filter are broken down into four (M = 4) polyphase filters. The coefficients of the FIR filter are shown at the top.

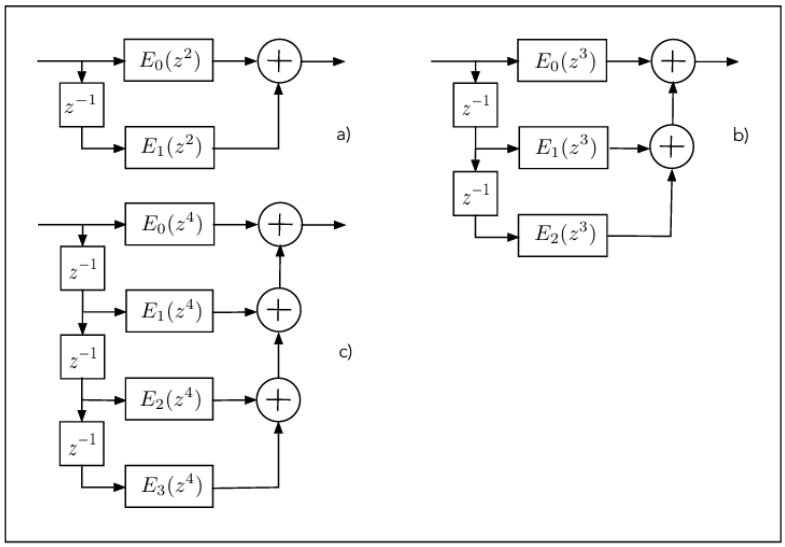

Figure 104 – Polyphase decomposition with M = 2, 3 and 4 [31]

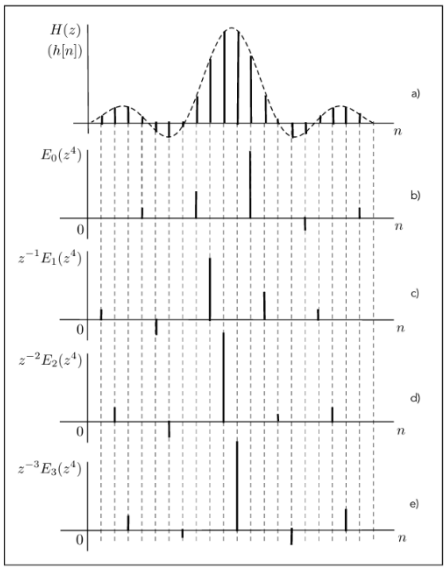

Figure 105 – Coefficients of the polyphase decomposition with M = 2, 3 and 4 [31]

Decimation with filter in polyphase realization:

In the area of multi rate signal processing there are several equivalent structures. One of these, which is important for the decimation, is shown in Fig. 106. In order to use this equivalence in a decimation with factor M, the FIR filter of the decimation must be decomposed into M branches. [32]

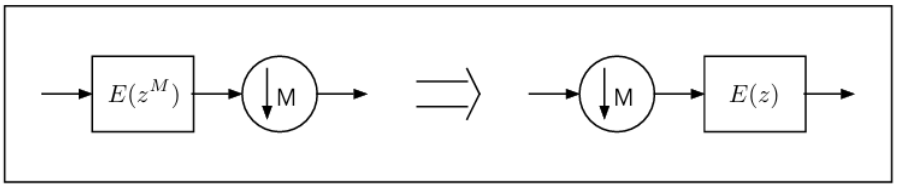

Figure 106 – Decimation equivalent structure in multi rate systems [32]

An FIR filter can be implemented as a parallel connection of $M(L)$ polyphase components, which are added together at the output. A polyphase component is usually implemented in the direct transversal form. Fig. 107a shows a decimator composed of the cascade of an FIR filter implemented as a parallel connection of M polyphase branches, and factor-of-M down-sampler. Here the arithmetic operations in the polyphase branches are to be performed at the input sampling rate, i.e. at the higher sampling rate of the system. Instead of down-sampling at the filter output, one can shift the down-sampling operation into the polyphase branches before the output adders. This modification opens the opportunity to arrive to the

efficient implementation of Fig. 107b. In the structure of Fig. 107b, the down-sampling-by-M occurs at the inputs of the polyphase components and filtering is performed at the sampling rate $F_x\ /M$. The overall computational complexity of the decimator is reduced by M. [31]

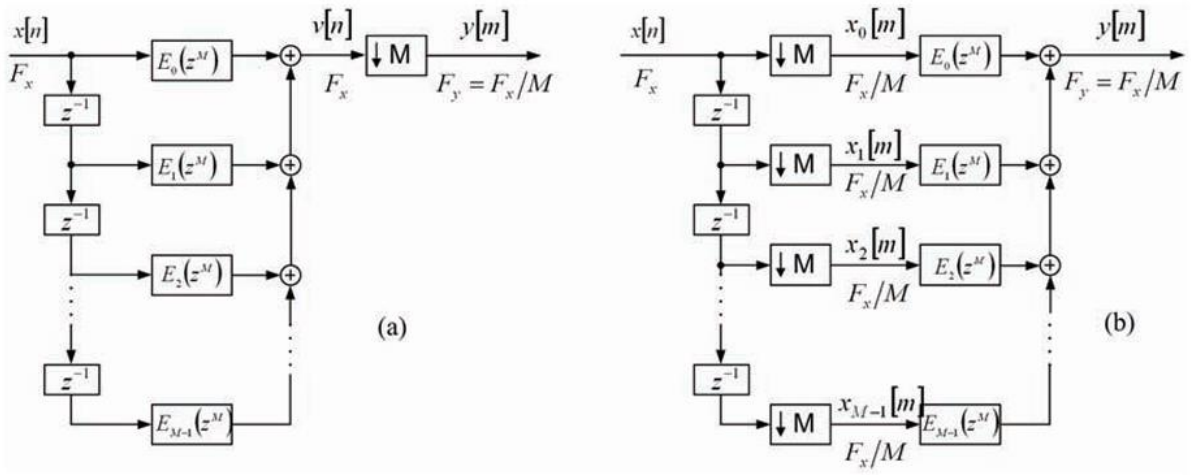

Figure 107 – Polyphase decomposition of a decimator [31]

Polyphase decomposition in this project:

After using the transposed implementation of FIR filters to improve the performance by introducing pipelining, polyphase decomposition is used to improve the performance even more by introducing parallel operation and by decreasing the overall power consumption.

Fig. 108 shows the Simulink model of the polyphase decomposition for the comb filter used in decimation. Fig. 109 shows the first two polyphase components E_0 and E_1.

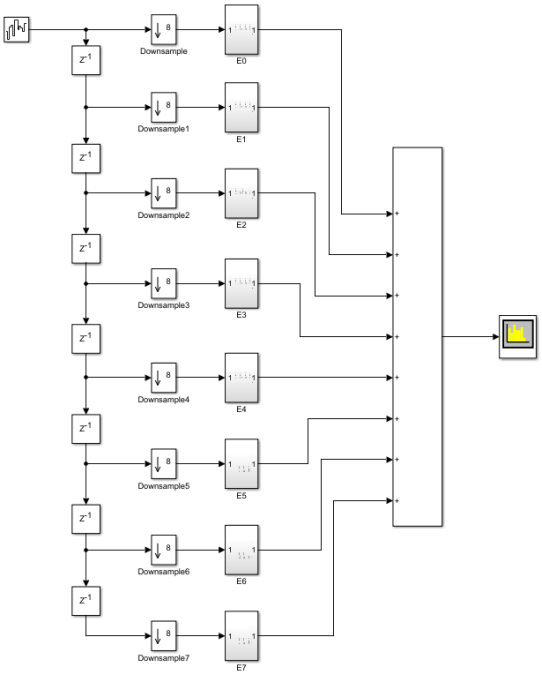

Figure 108 – Simulink model of the polyphase decomposition of the comb filter used in decimation

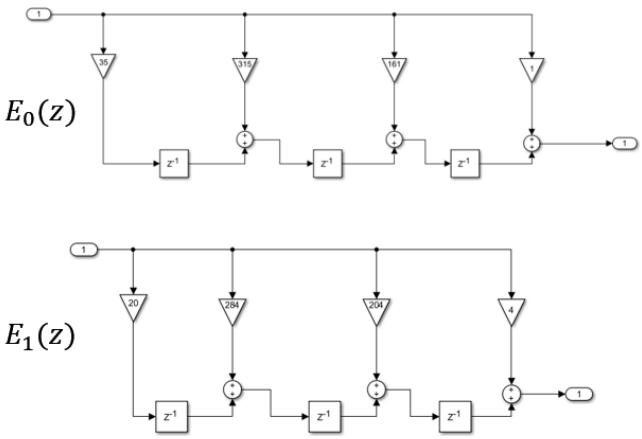

Figure 109 – Simulink model of the first 2 polyphase components of the comb filter used in decimation

Fig. 110 show the Simulink model of the polyphase decomposition for the half band filter used in decimation. Fig. 111 shows the two polyphase components E_0 and E_1.

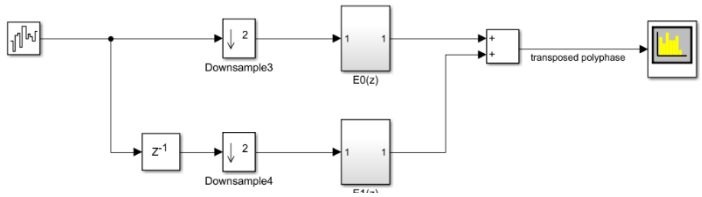

Figure 110 – Simulink model of the polyphase decomposition of the half band filter used in decimation

$E_0(z)$

$E_1(z)$

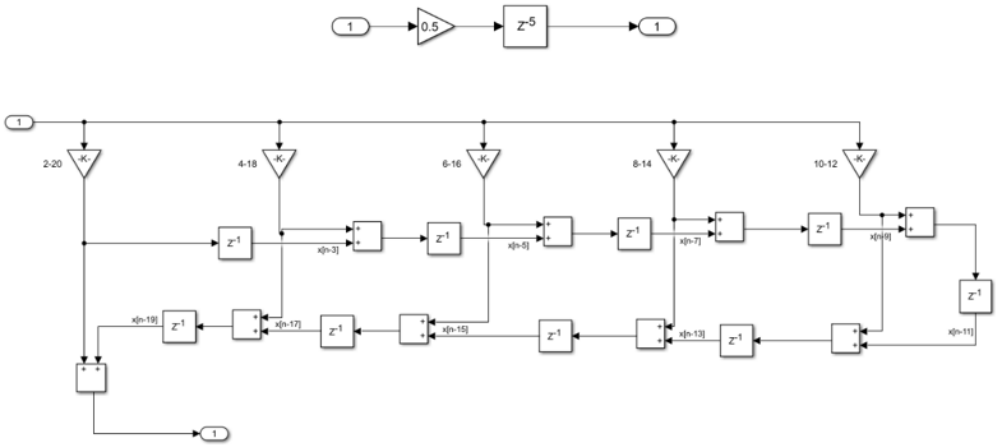

Figure 111 – Simulink model of the polyphase components of the half band filter used in decimation

Final Decimation Filter simulation

Fig. 112 shows the response of the decimation filter after cascading the two filters.

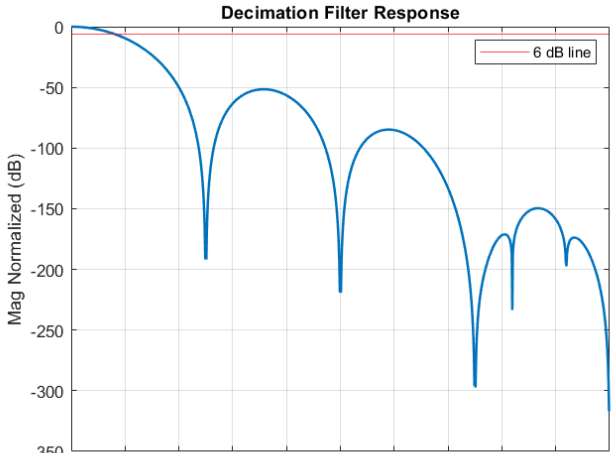

Figure 112 – Frequency response of the decimation filter

Fig. 113 and 114 show a Simulink model of the decimation filter and its frequency response. The frequency response is similar to the one from the MATLAB simulation in Fig. 38.

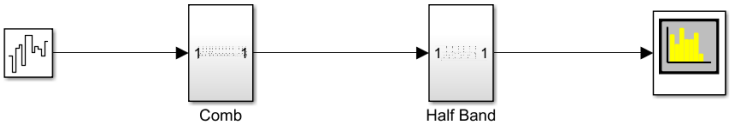

Figure 113 – Simulink model of the decimation filter

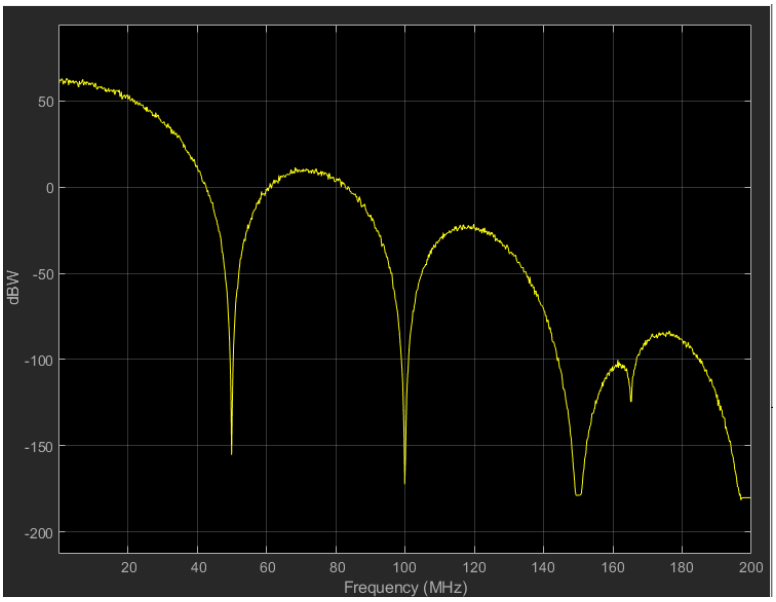

Figure 114 – Frequency Response of the Simulink model of the decimation filter

To ensure the functional correctness of the decimation filter (decimating the signal by 16) a simulation test is made in Simulink. The input signal is a sine wave with a frequency of $160\ Hz$ and the output signal is a sine wave with a frequency of $160/16 = 10\ Hz$ as shown in Fig. 115 and 116. The normalization needed for the decimation filter is $1/4096$ which is the sum of the coefficients of the comb filter.

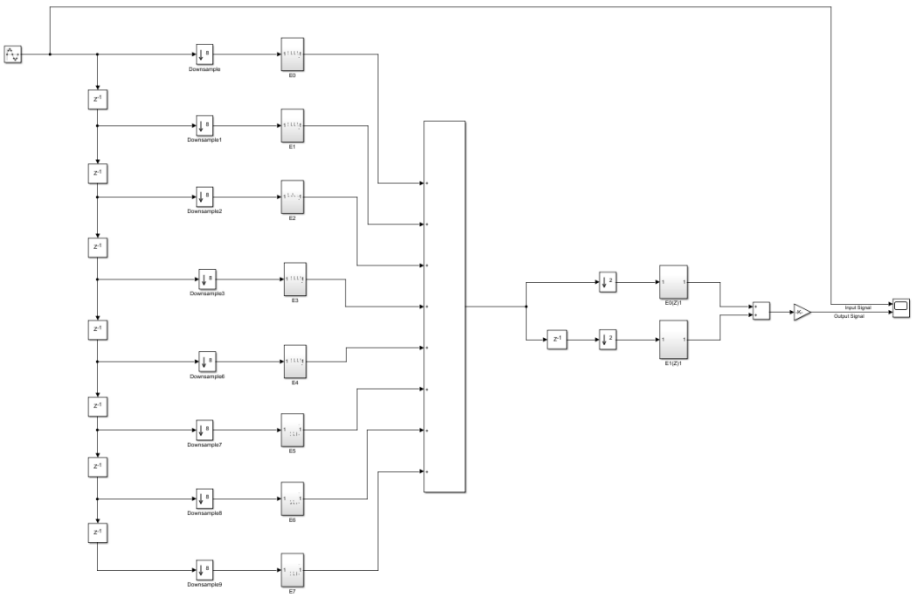

Figure 115 – simulation test in Simulink

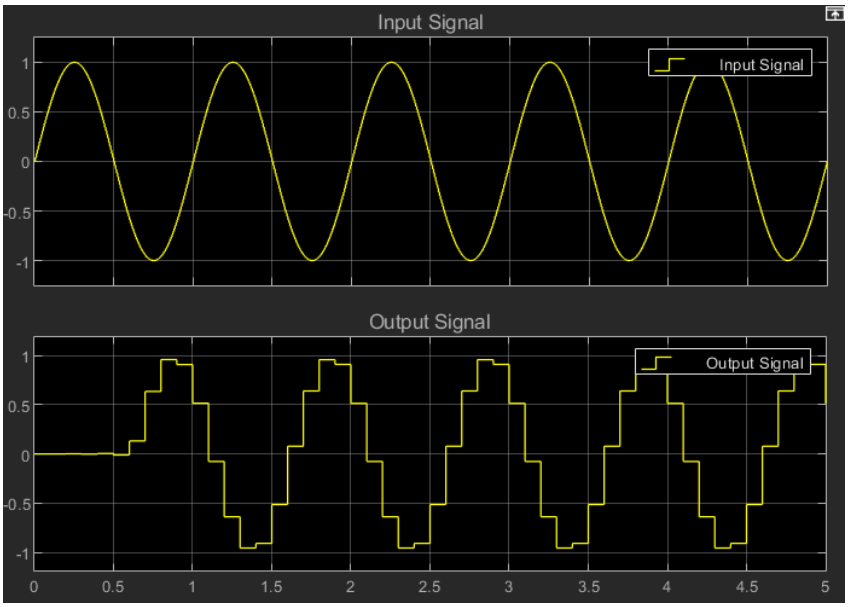

Figure 116 – Input and Output signals

4.3 Transmitter Model

Following the detailed analysis and design of the receiver model and its constituent blocks, a corresponding **transmitter model** has been proposed to implement **digital delay-and-sum beamforming**. This approach enables spatial filtering and directional transmission by digitally controlling the phase and timing of the transmitted signals across antenna elements.

The **transmitter chain** consists of the following key processing blocks:

- **Complex Weight Multiplication (CWM – Transmitter Side):**
  Applies a phase shift to the complex baseband signal through multiplication by a complex coefficient $e^{j\theta}$, enabling directional control and beam steering.
- **Interpolation Filter:**
  Increases the sampling rate of the baseband signal prior to Delta-Sigma modulation. This is essential to support the high oversampling requirements and to reduce aliasing in the subsequent stages.
- **Digital Delta-Sigma Modulator (DDSM):**
  Converts the high-resolution interpolated signal into a high-frequency bitstream while shaping the quantization noise out of the signal band, facilitating efficient RF signal generation.
- **Digital Up-Converter (DUC):**
  Shifts the baseband signal to the desired intermediate or carrier frequency by applying digital mixing and additional interpolation as required. This prepares the signal for transmission over the air through the RF front-end.

Fig. 117 shows the proposed transmitter model that complements the receiver design and ensures full support for **digital beamforming** in Massive MIMO 5G systems, while maintaining efficiency, precision, and compatibility with Delta-Sigma-based bit-stream processing.

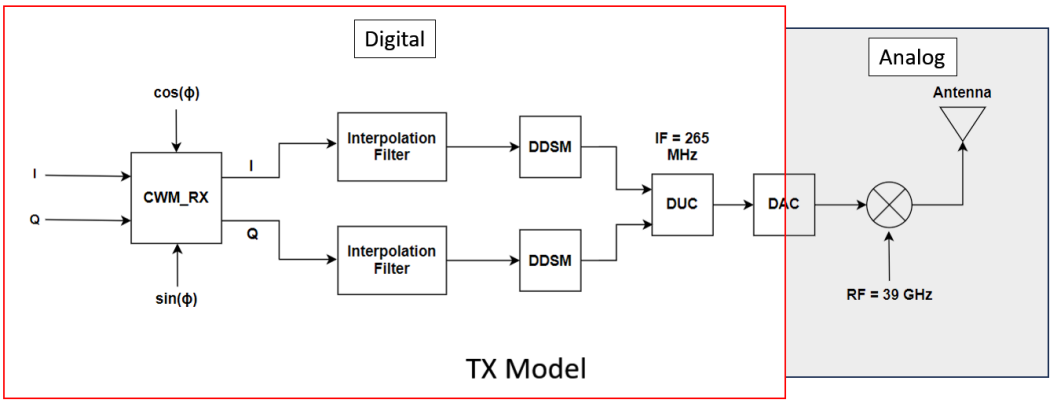

Figure 117 – Proposed Transmitter model

The following sections provide a detailed breakdown of each block within the digital domain of the proposed Transmitter. Each block is presented with the following structure:

- **Theoretical Background**: An overview of the underlying principles and signal processing concepts relevant to the block's function.
- **Proposed Design**: A description of the design approach adopted to meet the system requirements, including functional and architectural considerations.
- **Optimization**: Techniques applied to enhance performance, reduce complexity, or improve power and area efficiency.
- **MATLAB and Simulink Modeling**: Simulation and validation of the block's behavior using MATLAB and Simulink to ensure correct functionality prior to hardware implementation.
- **Final Design**: The finalized version of the block, prepared for integration into the overall system and ready for RTL implementation.

This structured approach ensures that each component of the receiver is rigorously analyzed, designed, and validated before moving forward in the development process.

4.3.1 Complex Weight Multiplication (TX):

The first block in the transmitter chain is the Complex Weight Multiplication (CWM) block, which performs the inverse operation of its counterpart in the receiver. This block is responsible for applying phase shifts to the complex baseband signal in order to achieve beam steering and directional transmission in MIMO antenna arrays.

The input to the CWM block is a complex signal of the form:

$$S = I + jQ \tag{81}$$

This signal is multiplied by a complex coefficient, or **weight**, typically expressed as a unit-magnitude phasor:

$$W = e^{j\theta} \tag{82}$$

This multiplication adjusts the phase of the signal without altering its amplitude, enabling precise control over the direction of the transmitted beam.

The operation can be rewritten as:

$$(I + jQ) * e^{j\theta} = (I + jQ) * (\cos(\theta) + jsin(\theta)) \tag{83}$$

Expanding the expression using complex multiplication yields:

$$I' = Icos(\theta) - Qsin(\theta)$$
$$Q' = Isin(\theta) + Qcos(\theta) \tag{84}$$

This defines the transformed in-phase (I') and quadrature (Q') components of the output signal. These equations form the basis of the **phase shifter block diagram**, which is a direct hardware interpretation of the CWM functionality. The block effectively rotates the input vector in the complex plane by an angle θ, facilitating directional signal transmission in the antenna array. Fig. 118 shows the block diagram of CWM in the transmitter chain.

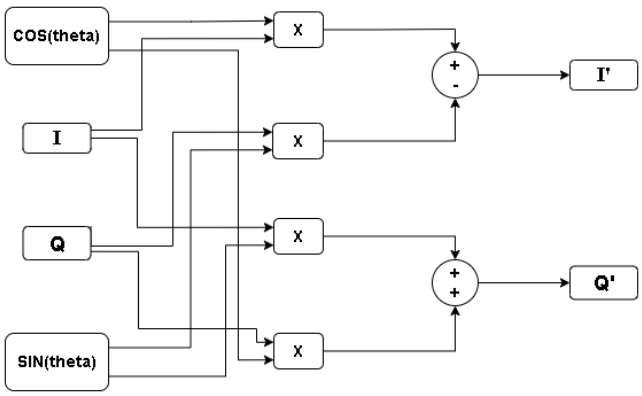

Figure 118 – CWM block diagram (Tx)

Complex Weight Multiplication (CWM) involves computing real-time trigonometric functions such as sine and cosine for phase rotation. In hardware, using its value as inputs or LUT-based implementations for these functions can consume significant area and power.

To reduce hardware complexity for optimization, CWM block is implemented using CORDIC (Coordinate Rotation Digital Computer) algorithm. CORDIC enables the computation of sine and cosine values using only shift-add operations, without the need for multipliers or LUTs.

Advantages of using CORDIC:

- Low-power and area-efficient.
- Scalable.
- Good accuracy for fixed-point implementation.

CORDIC:

The CORDIC (Coordinate Rotational Digital Computer) can break the basic functions down to operations of shift and addition or subtraction, which can be used to lay the foundation for the realization of complex logic.

CORDIC algorithm is the best choice to achieve the functions of transcendental functions such as trigonometric, inverse trigonometric, exponential function, logarithmic function since that the CORDIC algorithm is provided with a simple structure, and characteristic of saving resources and high efficiency.

CORDIC algorithm is an iteration algorithm and commonly used to calculate basic arithmetic functions. The principle of the CORDIC is to use the deflection of the angles associated with the cardinal number, rather than get the desired angle. Thus, the principle can be considered as a numerical approximation method of calculation. Because the fixed angle is related with cardinal number, the operations of computation include only shift and addition or subtraction. Therefore, it will cost less resources than conventional calculation methods, such as multiplication and division.

CORDIC algorithm is mainly used to solve the problem in two-dimensional vector rotation plane. It replaces the rotation operations with simple ones, such as shift and addition or subtraction. Through the algorithm we can divide the larger target rotation angle θ into several consecutive smaller deflection angles α_i, so as to achieve the process of rotation. The delay is mainly determined by the number of iterations and the time spent in each iteration stage. [35]

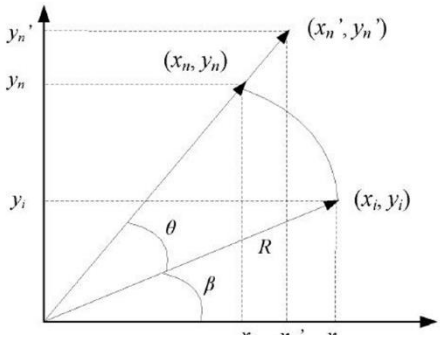

Figure 119 – CORDIC Algorithm Rotation [35]

From Fig. 119 we obtain a new vector (x_n, y_n) by rotation of the initial vector (x_i, y_i), the coordinates of the new vector can be expressed as:

$$x_n = \sqrt{(x^2 + y^2)}\cos(\theta + \beta) = \cos(\theta)\, x_i - \sin(\theta)\, y_i$$
$$y_n = \sqrt{(x^2 + y^2)}\sin(\theta + \beta) = \sin(\theta)\, x_i + \cos(\theta)\, y_i \quad (85)$$

Eq. (85) can be written in matrix form as:

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad (86)$$

Assuming $\tan(\alpha_i) = 2^{-i}$, the angle of rotation of each step $\alpha_i = \delta_i \tan^{-1}(2^{-i})$ can be approximated as:

$$\theta = \sum_{i=0}^{N} \sigma_i \alpha_i + \varepsilon = \sum_{i=0}^{N} \sigma_i \tan^{-1}(2^{-i}) + \varepsilon \tag{87}$$

In Eq. (87), $\delta_i = \{1, -1\}$, the sign of δ_i determines the direction of rotation, which would be close to the target vector if $\delta_i = 1$, else would be close to the opposite direction if $\delta_i = -1$. Then the remaining angle after each rotation can be expressed as: $Z_{i+1} = Z_i - \delta_i \alpha_i$, where $\delta_i = sign(Z_i)$.

Assuming that we could complete the rotation angle θ by N times of iterations, then the rotation process may be represented by Eq. (88).

$$\begin{aligned} \begin{bmatrix} x_{N+1} \\ y_{N+1} \end{bmatrix} &= \prod_{i=1}^{N} \begin{bmatrix} \cos(\alpha_i) & -\sin(\alpha_i) \\ \sin(\alpha_i) & \cos(\alpha_i) \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \\ &= \prod_{i=1}^{N} \cos(\alpha_i) \prod_{i=1}^{N} \begin{bmatrix} 1 & -\tan(\alpha_i) \\ \tan(\alpha_i) & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \\ &= \prod_{i=1}^{N} \frac{1}{\sqrt{1+2^{-i}}} \prod_{i=1}^{N} \begin{bmatrix} 1 & -2^{-i} \\ 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \end{aligned} \tag{88}$$

Where $k = 1/\sqrt{1+2^{-i}}$, k_i would converge to a constant as while as the increasing of the number of iterations. The gain constant K of the rotational operation can be expressed as:

$$k = \prod_{i=1}^{N} k_i = \prod_{i=1}^{N} \frac{1}{\sqrt{1+2^{-i}}} \tag{89}$$

The value of K depends on the number of iterations N, and K was usually called focus constant or scaling factor. [35]

CWM design using CORDIC:

The CORDIC algorithm works by **rotating a vector** step-by-step to approach a given angle θ. Each rotation is pre-defined by arctangent values and performed using only **shift-add** operations.

For each iteration n:

$$x_{n+1} = x_n - d_n y_n 2^{-n} \tag{90}$$

$$y_{n+1} = y_n + d_n x_n 2^{-n} \tag{91}$$

$$z_{n+1} = z_n - d_n \cdot \tan^{-1}(2^{-n}) \tag{92}$$

where $X_o = 1,\ Y_o = 0,\ Z_o = \theta(input\ from\ CWM),\ d_n = \{1, -1\}$

After N iterations:

$$x_n \approx \cos(\theta) . K \quad (93)$$

$$y_n \approx \sin(\theta) . K \quad (94)$$

$$k \approx 0.60725 \quad (95)$$

Fig. 120 shows the SIMULNK model of the CWM block showing the input I and Q signals and the generated I' and Q'. Fig. 121 shows applying the CODRIC as the approximation method for the trigonometric functions.

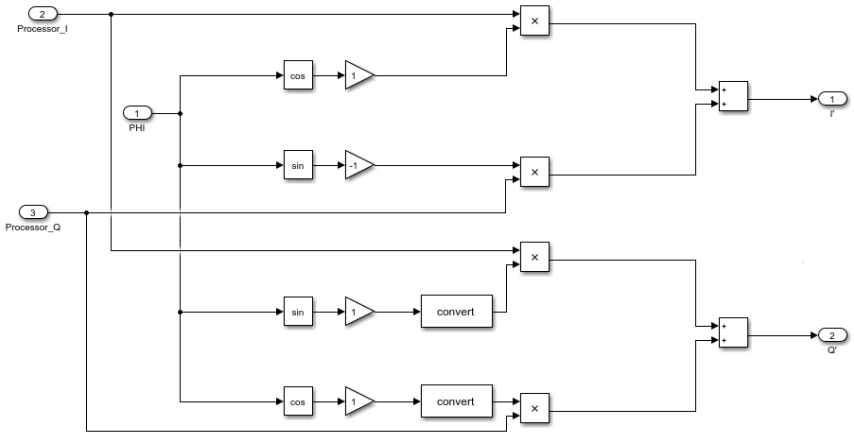

Figure 120 – Simulink model of CWM

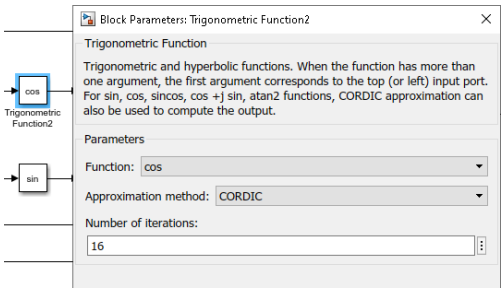

Figure 121 – CORDIC in CWM

4.3.2 Interpolation

Sampling rate conversion systems are used to change the sampling rate of a signal. The process of sampling rate decrease is called decimation, and the process of sampling rate increase is called interpolation. Two devices, the down-sampler and the up-sampler, are elements that change the sampling rate of the signal. The drawback of the down-sampling is the aliasing effect, whereas the up-sampling produces the unwanted spectra in the frequency band of interest. Decimation has to be performed in such a way as to avoid the effects of aliasing. [31]

Interpolation requires the removal of the images, see Fig. 122. This means that the factor of L interpolation has to be implemented in two steps:

(1) Up-sampling of the original signal by inserting L-1 zero-valued samples between two consecutive samples.

(2) Removal of the L-1 images from the spectrum of the up-sampled signal.

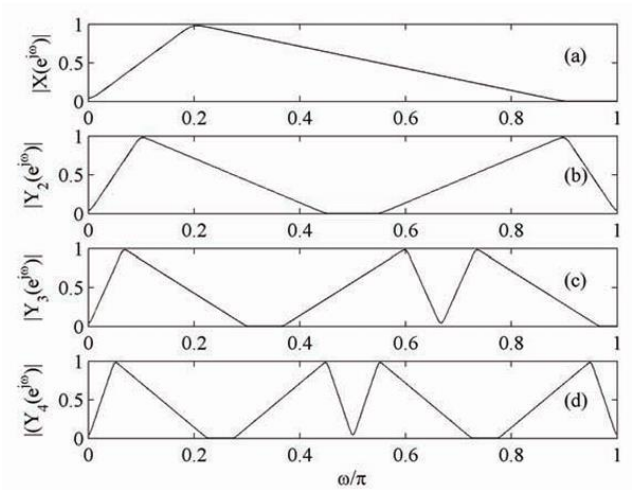

Figure 122 – Frequency domain of an up sampled signal

By interpolation, values are interpolated between the sampled values of a signal as if the original time-continuous signal had been sampled with a higher sampling frequency. It is assumed that the sampled values of the time-continuous original signal before interpolation corresponded to the sampling theorem and thus contain all the information that enables interpolation. In concrete terms, this means that $f_s \geq 2f_{max}$. The sampling frequency of the continuous-time signal was designated with f_s and f_{max} is the maximum frequency in the spectrum of this signal. [32]

Interpolation of band limited lowpass signals:

In Fig. 123a the structure of the classical interpolation is shown. The input signal $x[nT_s]$, which fulfills the conditions shown above, is brought through upsampling to a frequency $f_s' = Lf_s$, where L is the interpolation factor (an integer). The upsampling adds $L-1$ zero values between the original sample values, so that the period T_s is divided into L equal intervals of the size $T_s' = T_s/L$. This gives the places where the FIR interpolation filter adds the interpolated values.

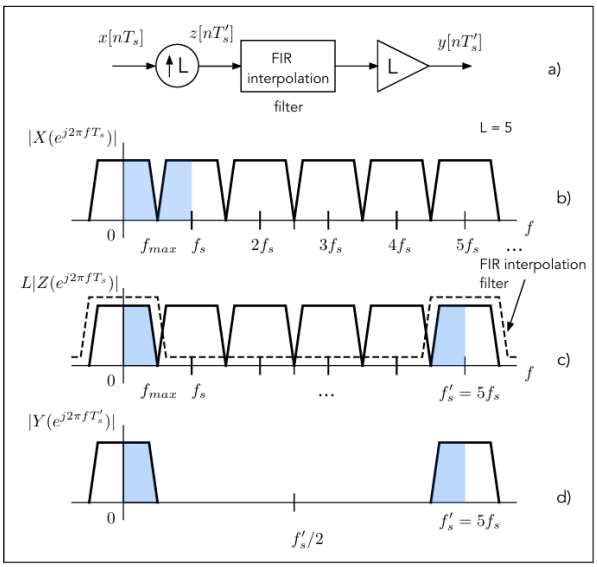

Figure 123 – The Interpolation process (L=5) [32]

In Fig. 123b the spectrum of the original signal is shown, it repeats itself periodically until $f \rightarrow \infty$. Assuming that $L = 5$, the figure shows the spectrum up to this frequency. The up-sampling now results in the signal $z[nT_s']$ with a discrete sampling period $Ts' = Ts/L$ and a spectrum that is proportional to the spectrum of the signal $x[nT_s]$, as shown in Fig. 123c. In order to obtain a signal with a spectrum as in Fig. 123d, which corresponds to the time-continuous original signal sampled with a sampling frequency f_s', a low-pass filter with an amplitude response that is sketched with a dashed line in Fig. 123c must be used. [4]

Proposed design

For this project the needed interpolation factor is 16 (same as the decimation) and the interpolation filter proposed consists of two stages and they are the same two stages in decimation but reversed:

1- Half band filter with decimation factor of 2.

The half band filter has a cut off frequency at $\pi/2$ which means large passband region so it's used first because the passband region of the comb is very small which can destroy the signal.

2 - Comb filter with decimation factor of 8:

The comb filter acts as a LPF FIR filter that is used for the removal of the images after up sampling.

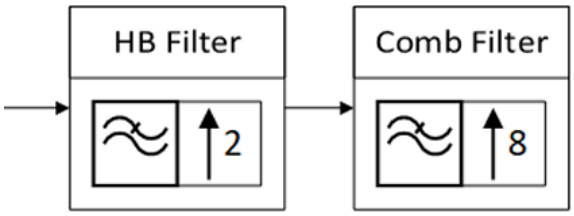

Figure 124 - The proposed Interpolation Filter

The main advantage of this architecture is that it is the same architecture and the same design of the decimation filter so the same comb and half band filters will be used in interpolation same as in decimation.

Optimization:

Transposed FIR filters - Same transposed comb and half band filters will be used in interpolation.

Polyphase decomposition - Same concept of Polyphase decomposition will be used in interpolation same as decimation.

Interpolation with polyphase realization:

An equivalent structure from the field of multi-rate signal processing, which is shown in Fig. 125, can be used advantageously for the interpolation. The decomposition into polyphase

filter structures was introduced in connection with decimation and is also valid for interpolation.

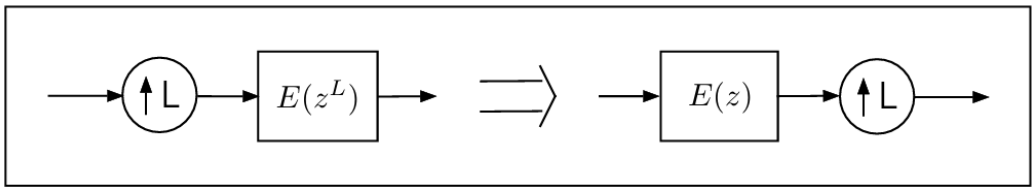

Figure 125 – Interpolation equivalent structure in multi rate systems [4]

The parallel polyphase structure composed of L polyphase components can be used to provide an efficient implementation of a factor of L interpolator. Fig. 126a shows the interpolator consisting of a factor of L down-sampler and an FIR filter realized in the polyphase form. In the interpolator structure of Fig. 126a, the up-sampling precedes filtering, and according to this, filtering in the polyphase components is performed at the higher sampling rate. The structure of Fig. 126a can be modified to the more efficient structure shown in Fig. 126b. The positions of up-samplers and polyphase components are interchanged, and filtering in the polyphase branches is to be performed at the lower sampling rate. [31]

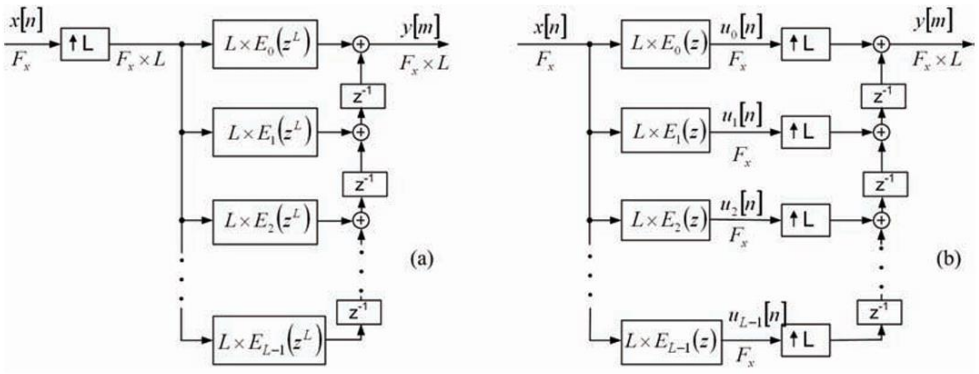

Figure 126 – Polyphase decomposition of an interpolator [31]

Polyphase decomposition in this project:

Fig. 127 show the Simulink model of the polyphase decomposition for the comb filter used in interpolation. Fig. 128 shows the first two polyphase components E_0 and E_1.

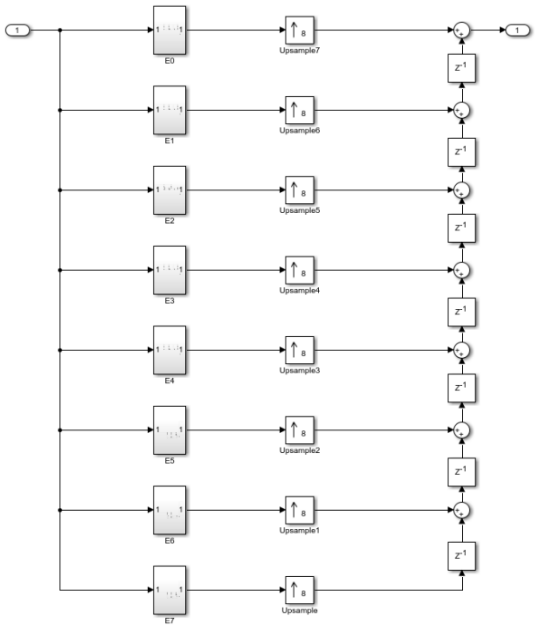

Figure 127 – Simulink model of the polyphase decomposition of the comb filter used in interpolation

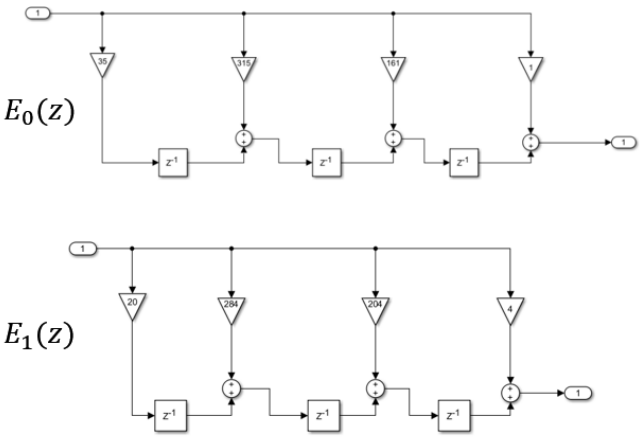

Figure 128 – Simulink model of the first 2 polyphase components of the comb filter used in interpolation

Fig. 129 show the Simulink model of the polyphase decomposition for the half band filter used in interpolation. Fig. 130 shows the two polyphase components E_0 and E_1.

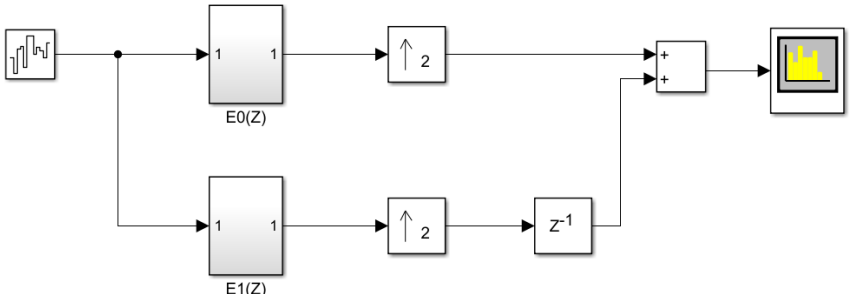

Figure 129 – Simulink model of the polyphase decomposition of the half band filter used in interpolation

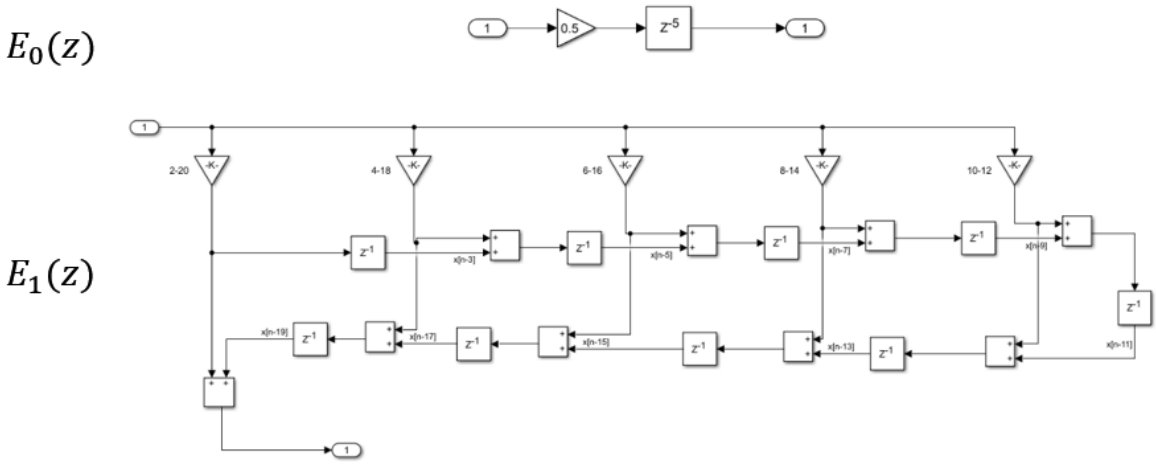

Figure 130 – Simulink model of the polyphase components of the half band filter used in interpolation

Final Interpolation Filter design

Fig. 131 shows the response of the Interpolation filter after cascading the two filters.

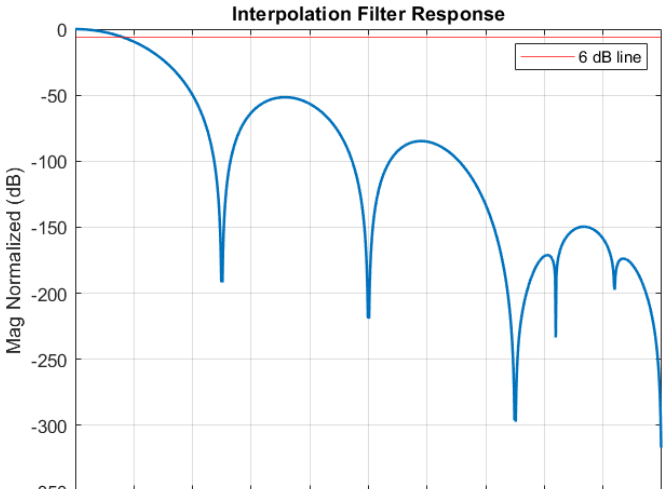

Figure 131 – Frequency response of the interpolation filter

Fig. 132 and 133 show a Simulink model of the interpolation filter and its frequency response. The frequency response is similar to the one from the MATLAB model in Fig. 15.

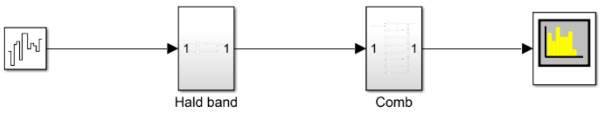

Figure 132 – Simulink model of the interpolation filter

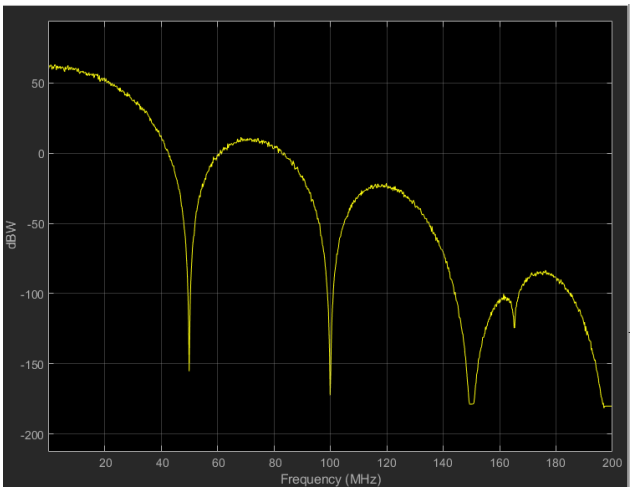

Figure 133 – Frequency Response of the Simulink model of the interpolation filter

To ensure the functional correctness of the Interpolation filter (interpolating the signal by 16) a simulation test is made in Simulink. The input signal is a sine wave with a frequency of $10\ Hz$ and the output signal is a sine wave with a frequency of $10 \times 16 = 160\ Hz$ as shown in Fig. 134 and 135. The normalization needed for the interpolation filter is $1/256$ which is the sum of the coefficients of the comb filter.

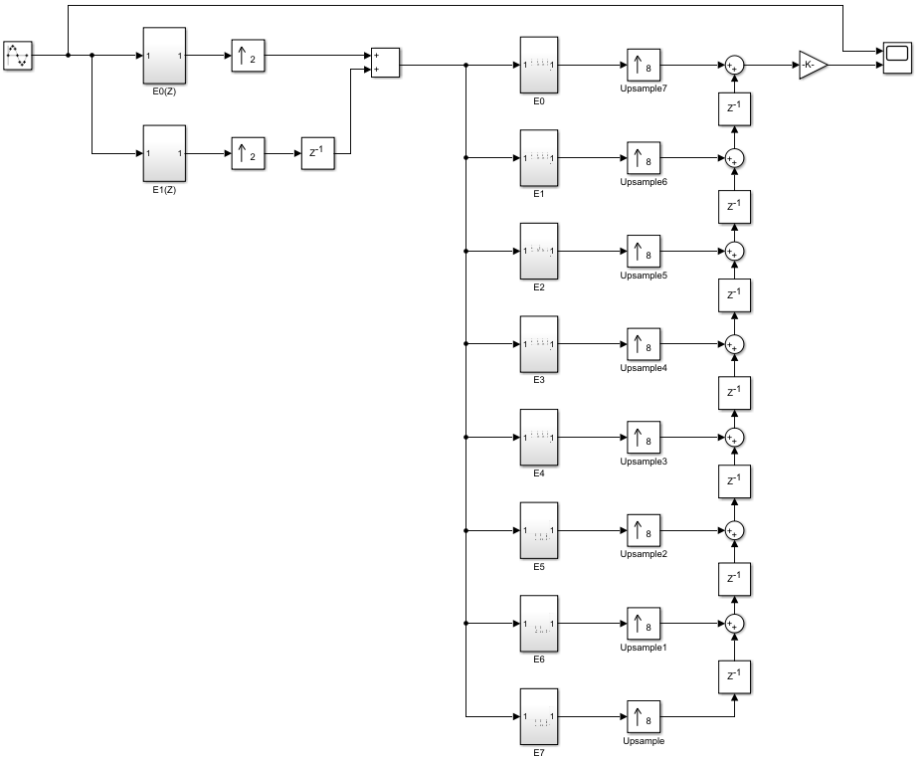

Figure 134 – simulation test in SIMULINK

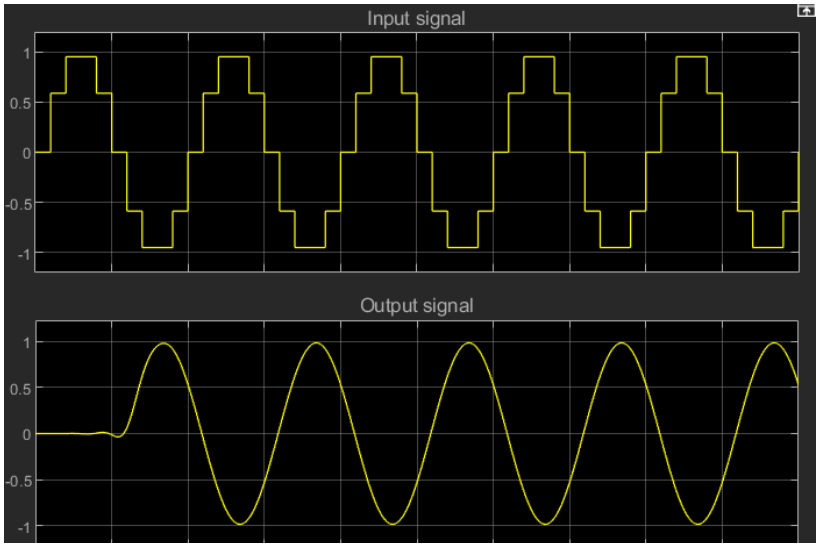

Figure 135 – Input and Output signals

4.3.3 DDSM

The Digital Sigma-Delta Modulator (DSM) is a fundamental building block in modern digital transmitter architectures performing two core functions. Firstly, the DSM acts as a quantizer by converting a high-resolution digital input into a lower-resolution digital output reducing

the signal's bit count. This reduction in bit-depth greatly simplifies the design of the analog DAC stage that follows the transmitter.

Secondly, it performs noise shaping by pushing most of the quantization noise power to higher frequencies, outside the signal bandwidth instead of spreading the noise uniformly across the entire frequency band. As a result, after the signal passes through a low-pass filter in the RF domain, the in-band quantization noise is significantly reduced, enhancing the overall signal-to-noise ratio (SNR) within the desired frequency band and resulting clearer signal.

The Digital Delta–Sigma Modulator (DDSM) shown in Fig. 136 is a discrete-time deterministic dynamic system with a digital input and a digital output. The DDSM receives an n_o digital input sequence and delivers an m bit digital output sequence, as shown. Generally, m is significantly less than n_o.

Figure 136 - Block diagram of the DDSM with n-bit input x[n] and m-bit output y[n] [37]

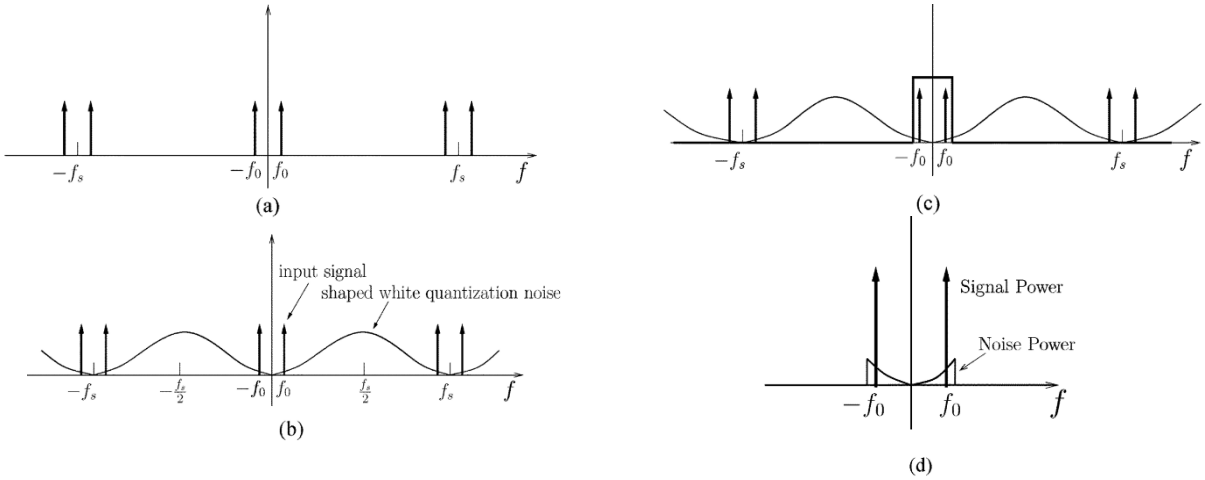

Figure 137 - Spectra in a DDSM with a sinusoidal input.
(a) Power spectrum of the input tone. (b) Power spectrum of the output sequence.
(c) The LPF is applied to the DDSM output. (d)The LPF output spectrum (enlarged). [37]

Fig. 137 illustrates the operation of the DDSM in the frequency domain. Assume that we have applied to the input of the modulator a high-resolution low-frequency tone at f_o encoded over n_o bits with the spectrum given in Fig. 137a. As shown in Fig. 137b, the DDSM delivers a low-resolution sequence to the output whose spectrum contains the original spectrum of the input signal.

Note that the spectrum of the low-resolution output sequence contains the full information of the high-resolution input sequence. Ideally, the resulting quantization error is whitened

and filtered by the DDSM so that its power is moved away from the signal through a process called noise shaping.

The shaped quantization error (noise) appears in the output spectrum as shown in Fig. 137b. Note that the power of the quantization noise is concentrated away from f_o and around f_o / 2, thereby maximizing the signal-to-noise ratio close to dc. If the output signal of the DDSM is applied to an ideal continuous-time low-pass filter (Fig. 137c) that passes only the low-frequency content of the modulator output, then the original signal can be recovered from the low-resolution output signal with high signal-to-quantization noise ratio.

Fig. 137d shows the spectrum at the output of the low-pass filter. Note that the signal power of the filtered output is much higher than the noise power content. DDSMs may be implemented in several different ways. We will investigate the Error Feedback (EFB) and the Multistage noise Shaping (MASH). [37]

EFB

The first possible architecture is a first-order EFB DSM, This DSM is actually an integrator whose output is quantized, the p-bit integrator is split into m MSBs that are sent forward to the DAC, while p-m LSBs are sent back as feedback into the integrator. This feedback is also referred to as the negative quantization error term $(-er)$. The output, $Y(z)$ can then be written as

$$Y(z) = X(z) + (1 - z^{-1})E(z) \tag{96}$$

where, $E(z)$ is the quantization error introduced at the output.

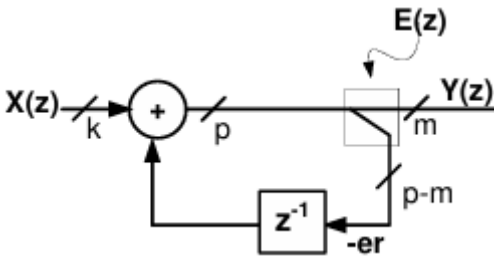

Figure 138 - A first-order EFB [2] DSM

This DSM is said to be stable as long as the integrator does not overflow. Being a first-order system, this is a stable system. However, a first-order EFB by itself is hardly used because it does not provide sufficient noise-shaping to achieve a high SQNR and suffers from limit cycles or idle tones in the output spectrum. To improve the achieved SQNR, a higher-order NTF function is used i.e. for an nth-order modulator, Eq. (97) can be rewritten as

$$Y(z) = X(z) + (1 - z^{-1})^n E(z) \tag{97}$$

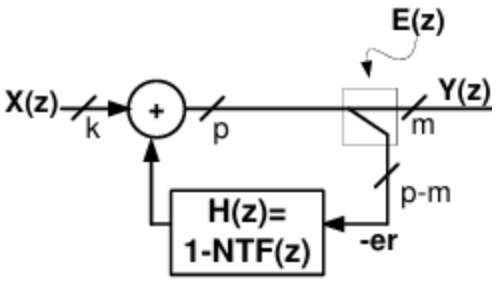

Figure 139 - An n^{th} order EFB DSM [38]

As the order increases, the number of adders in the critical path increases, thus limiting the maximum frequency of operation. If the multiplication coefficients are notpowers-of-2 then there is additional computational overhead, which further limits the speed. For a third order EFB,

$$H(z) = 3z^{-1} - 3z^{-2} + z^{-3} \tag{98}$$

Since, all the coefficients are not powers-of-2, they also have to be expressed as a sum of powers-of-2 to avoid any multipliers in the design e.g. $3 = 2^1 + 2^0$.
Although the order is third, the critical path becomes four adders as shown in Fig. 140.

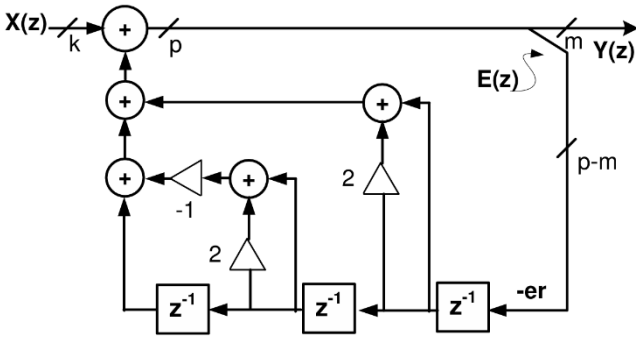

Figure 140 – A third-order EFB DSM with four adder critical path [38]

The length of critical path thus depends on the coefficient of the multiplier and the order of the modulator. Hence, this architecture is not well-suited for a high-speed implementation due to a large critical path. It can be noted that no pipeline stages (z^{-1}) between the adders are allowed as this alters the transfer function.

MASH

The second possible architecture is MASH architecture which has high potential for high-speed implementation. It consists of a cascade of individual EFB DSMs. The MASH architecture is shown in Fig. 141 wherein the error term, -er generated in each stage becomes the input of the next stage. The outputs then undergo a final processing (error cancellation) to achieve the final output. The individual DSMs can be of any order and the overall DSM order is the total sum of the order of all the stages.

In order to understand the MASH operation, consider an example MASH consisting of two stages where each stage is a simple first-order EFB DSM. Then, the input-output relations for the two stages can be written as

$$Y_1(z) = X(z) + (1 - z^{-1})\ E_{r1}(z) \tag{99}$$

$$Y_2(z) = -E_{r1}(z)(z) + (1 - z^{-1})\ E_{r2}(z) \tag{100}$$

Multiplying Eq. (99) by $(1 - z^{-1})$ and adding to Eq. (100)

$$Y(z) = X(z) + (1 - z^{-1})\ E_{r2}(z) \tag{101}$$

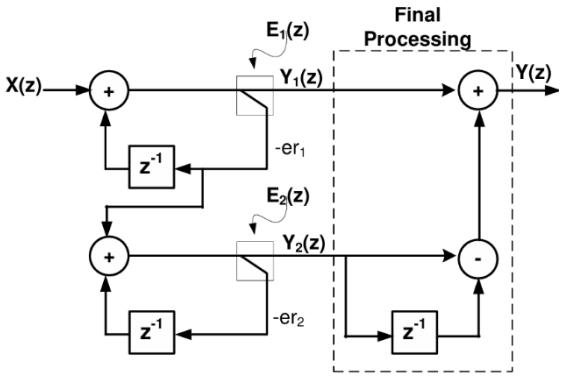

Figure 141 – A MASH 1-1 DSM [38]

It can be seen that the feedback path is confined only within each DSM or integrator. The path between the two DSMs is a forward path and hence can be optionally pipelined for a higher speed. Similarly, the final processing also consists only of forward paths only and hence can be optionally pipelined if required. Thus, the critical path of the MASH1-1can be restricted to only one adder only. This is shown in Fig. 142.

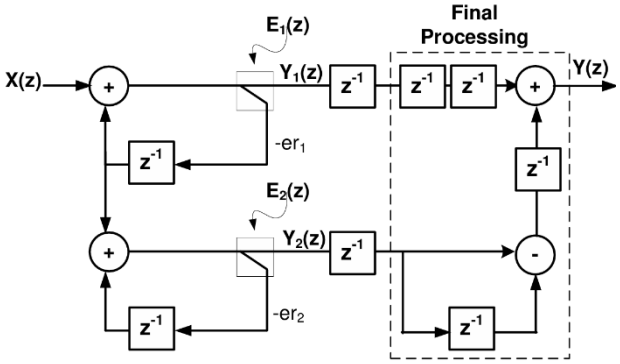

Figure 142 – A pipelined MASH 1-1 DSM with only one adder critical path [38]

This is an improvement over the EFB architecture with only one adder delay in the critical path. This scalability property of MASH makes it very attractive for high-speed implementations.

The MASH consisting of first order DSMs also offers some more practical advantages during the design. Only a first order DSM is required to be designed which can be instantiated multiple times depending upon the order. Secondly, the possibility of adding pipelines between the stages can be beneficial for timing closure. [38]

Proposed Design

After investigating some of the possible architectures, a 1-1-1 pipelined MASH architecture was chosen to be implemented as it is scalable and suitable for high frequency designs. Fig. 143 shows the implementation of the design on Simulink to model the block for testing and integration in the system.

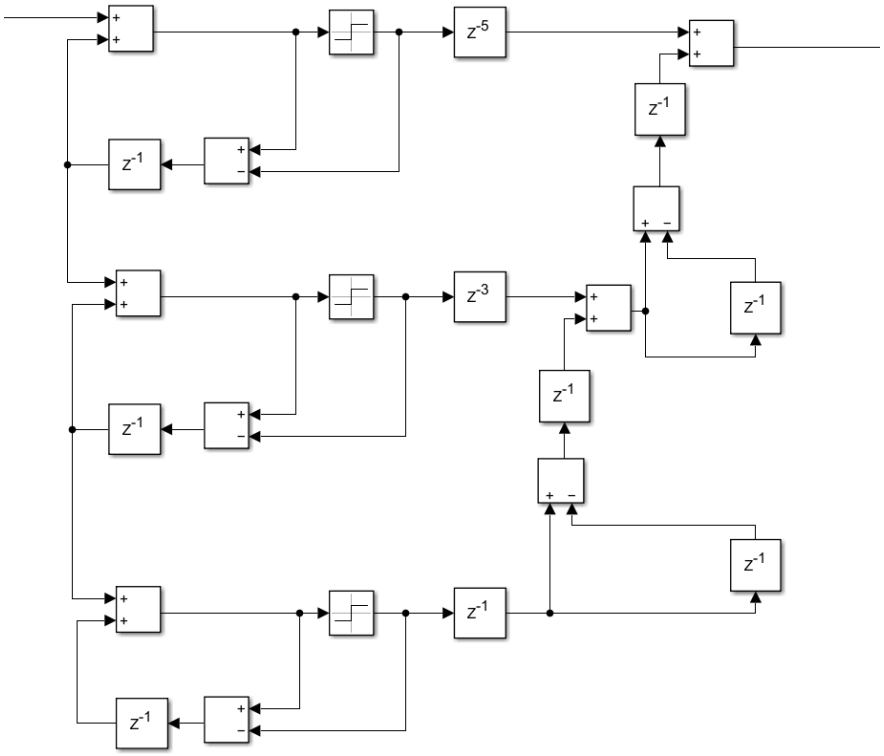

Figure 143 – Simulink model of the designed MASH 1-1-1 DSM

Testing is performed by applying a low-frequency single-tone input to the system and observing the output frequency response to verify that it conforms to the expected behavior of the DDSM.

The signal-to-noise ratio (SNR) around the input frequency is also calculated to confirm that the signal is not adversely affected by the noise-shaping process.

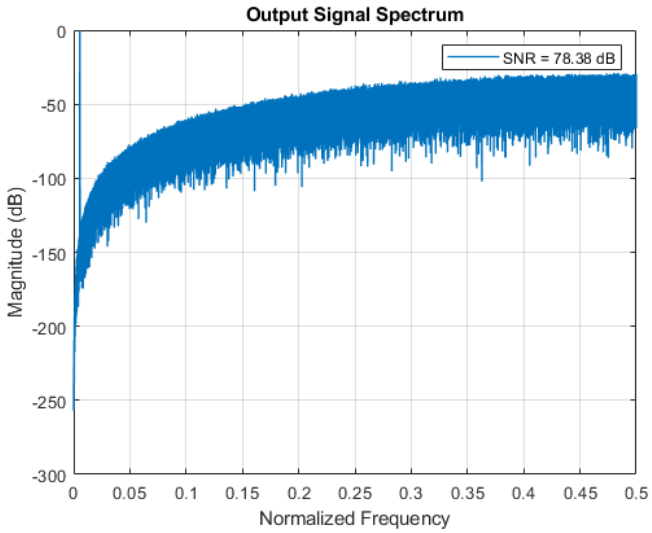

Figure 144 – Frequency response and SNR of the 1-1-1 MASH

As shown in Fig. 144, the frequency response of the system matches the expected behavior of the DDSM, shaping the quantization noise away from the signal frequency. The calculated SNR value also verifies that the desired signal can be effectively recovered through a low-pass filter with high SNR, as illustrated in Fig. 137d.

4.3.4 Digital up converter

A digital upconverter (DUC) accepts a complex I/Q waveform as its input. The DUC then converts this complex waveform to a real waveform with the same frequency content but centered at a higher IF center frequency. In other words, the baseband signal is "upconverted" to IF, and this up conversion is done digitally in DUC firmware.

The DUC is a set of algorithms implemented in an FPGA. Two main operations are performed by these algorithms: resampling and up conversion. The FPGA resamples the baseband I/Q waveform from the given I/Q rate to the digital-to-analog converter (DAC) sample rate. The DUC then multiplies the resampled I/Q waveform by an internally-generated complex IF carrier. This multiplication produces a waveform with a center frequency that matches the IF carrier frequency and frequency content that matches the baseband I/Q waveform, and this upconverted waveform is then fed into the DAC. [36]

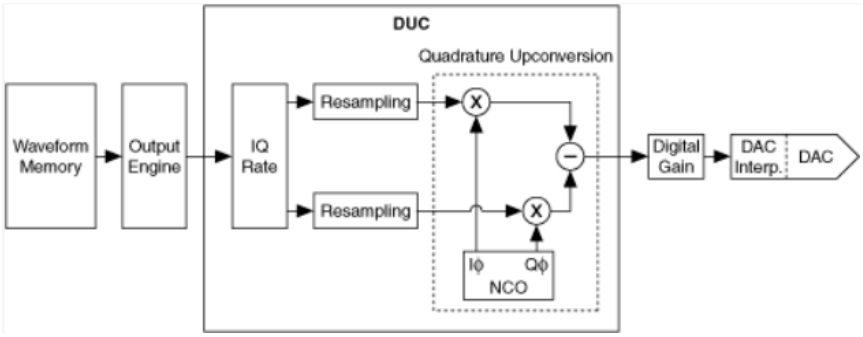

Figure 145 – DUC Architecture [36]

DUC design:

The in-phase (I) and quadrature (Q) components of each signal chain serve as inputs to the DUC. Here, they are modulated by being multiplied by sine and cosine signals at the IF.

The sampling time for the sine and cosine carriers is set at 4 times the carrier frequency. This approach simplifies the up-conversion process because the sampled sequence of sine and cosine alternates between {0, 1, 0, -1}, repeating cyclically. This pattern effectively translates multiplication into operations such as bypassing and inverting (multiplication by 1 or -1) and zeroing (multiplication by 0).

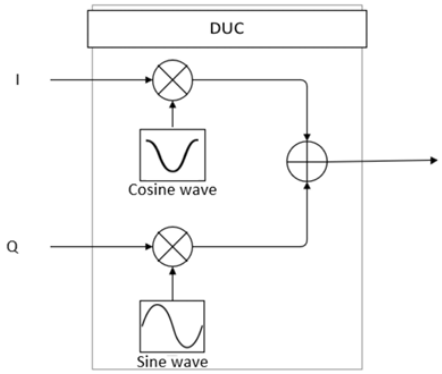

Figure 146 – DUC block diagram

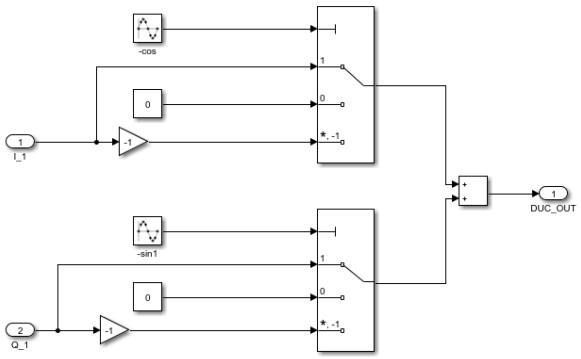

Figure 147 – Simulink model of the DUC

4.4 System integration & testing environment

To verify the functionality and coherence of all system blocks, a comprehensive testing environment was developed using Simulink. Figure 148 illustrates the implemented Simulink model, which comprises four complete transceiver chains. This setup enables validation that the transmitted signal can be correctly recovered after undergoing the entire transmit (Tx) and receive (Rx) processing chains.

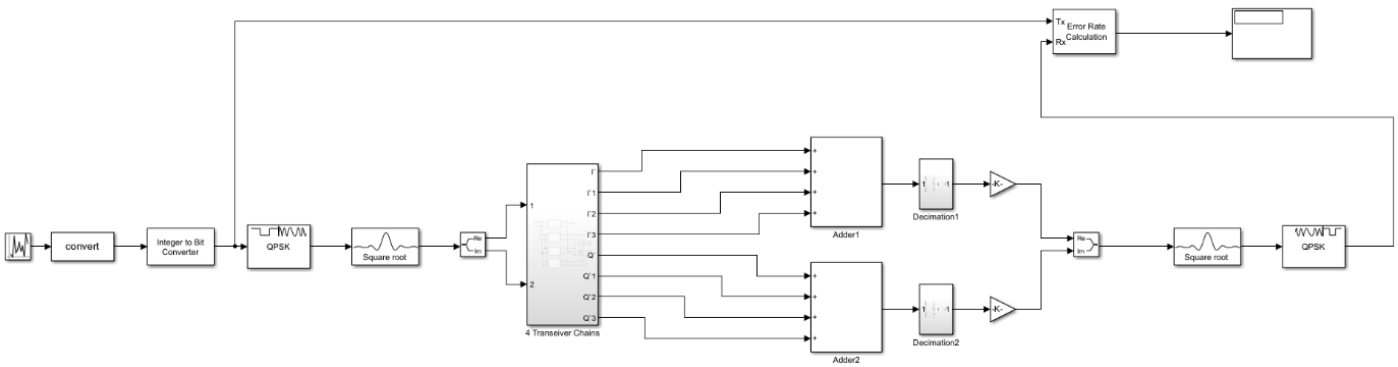

Figure 148 – Testing environment for the system

Random input data is generated with a uniform distribution between 0 and 3, using a fixed random seed (66) to ensure reproducibility across different channel signal-to-noise ratio (SNR) conditions. The generated symbols are first converted to a 2-bit fixed-point binary format, then processed through data type conversion and integer-to-bit conversion blocks to prepare them for modulation.

The QPSK modulator block modulates the input data using a Gray-coded constellation and a $\pi/4$ phase offset. A matching QPSK demodulator is placed at the receiver to perform the inverse operation. Between modulation and demodulation, the signal is shaped and filtered using raised cosine filters with a square root configuration, both at the transmitter and the receiver, to control bandwidth and inter-symbol interference.

After modulation and transmit filtering, the signal is fed into each of the four parallel transceiver chains. Within each chain, the signal undergoes Tx processing. The propagation channel is modelled as an additive white Gaussian noise (AWGN) channel, where the noise level is adjusted to simulate different SNR scenarios.

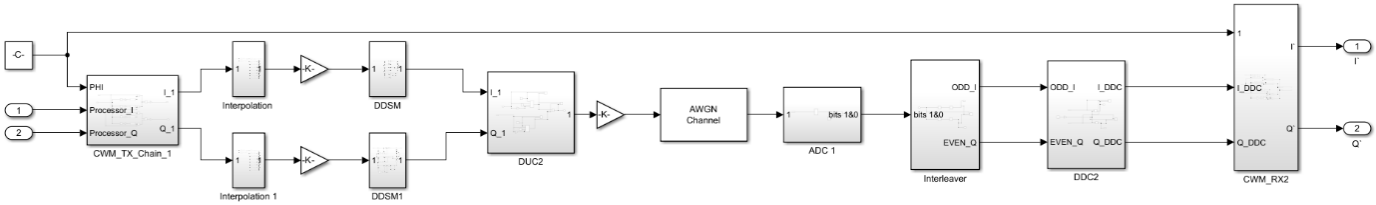

Figure 149 – Transceiver chain

At the receiver, the noisy signal is first digitized using a sigma-delta (ΣΔ) analog-to-digital converter (ADC) to produce a 1-bit digital output. The digitized signal is then processed by the Rx chain. The outputs of all four chains are summed and subsequently passed through two decimation filters to obtain the final in-phase (I') and quadrature (Q') components.

The I' and Q' components are combined and processed by the receive-side square root filter and QPSK demodulator. Finally, the recovered signal is compared with the original transmitted data using an error rate calculation block that compensates for the system delay.

To evaluate the impact of Signal-to-Noise Ratio (SNR) on Bit Error Rate (BER), the BER was computed across a range of SNR values. As expected, the results demonstrate that as the SNR decreases, the BER increases as seen in Fig. 150, 151 and 152, indicating degraded signal quality under higher noise conditions. This inverse relationship is consistent with theoretical expectations and validates the system's sensitivity to noise.

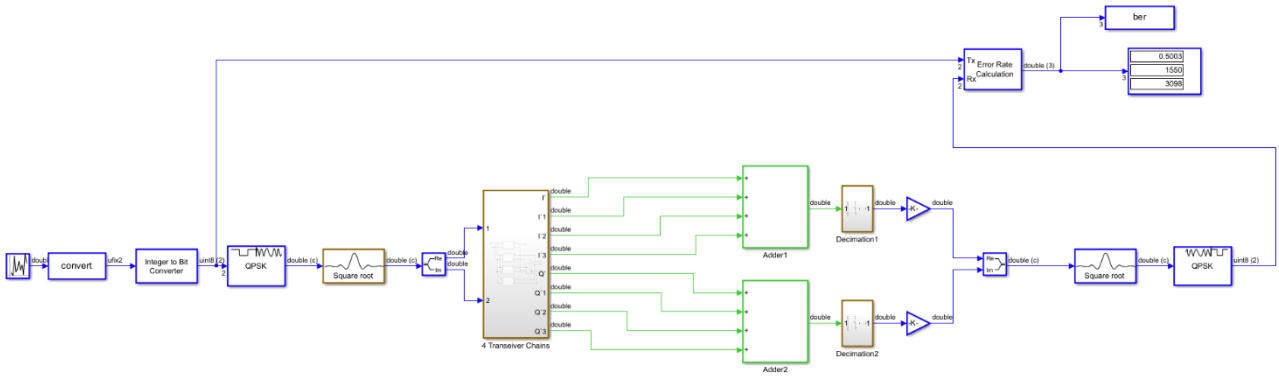

Figure 150 – BER calculation 1

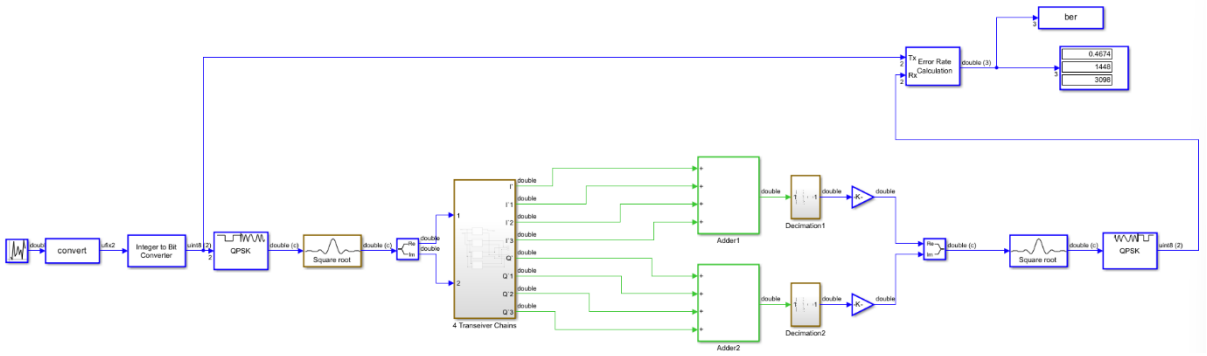

Figure 151 – BER calculation 2

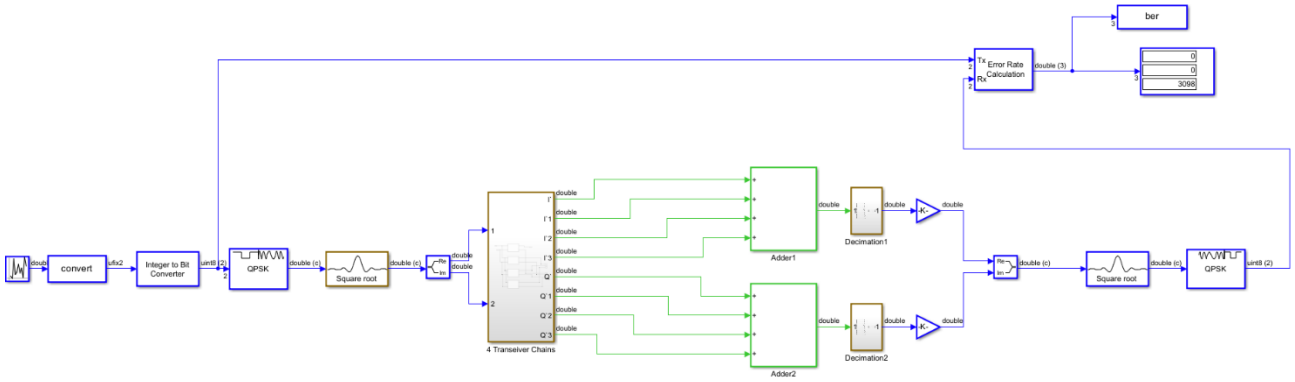

Figure 152 – BER calculation 3

Ch 5: Digital Design

The digital design phase begins with the transformation of the initial floating-point Simulink model, which was developed using fundamental digital signal processing blocks, into a fixed-point model suitable for hardware implementation.

To facilitate this conversion, data type conversion blocks are employed. These blocks convert floating-point signals into fixed-point representations, where both the integer and fractional bit widths are explicitly defined. Selecting an appropriate bit-width configuration is essential to balance numerical precision and hardware resource efficiency.

To determine the optimal bit-width while maintaining signal fidelity, a Signal-to-Quantization-Noise Ratio (SQNR) analysis is conducted. The SQNR is calculated using the following expression:

$$SQNR(dB) = 10\log_{10}\left(\frac{Signal\ Power}{Quantization\ Noise\ Power}\right)$$

The quantization noise is defined as the difference between the original floating-point signal and its corresponding fixed-point representation. A minimum SQNR threshold of 60 dB is adopted in this design to ensure that the fixed-point representation provides adequate accuracy for the intended signal processing tasks. If the SQNR falls below this threshold, the bit-width is adjusted accordingly to achieve the required precision.

This systematic approach ensures that the fixed-point model represents the original floating-point design while remaining suitable for RTL synthesis and hardware realization. Fig.153 shows an example of the data converter used across all the floating-point model Simulink.

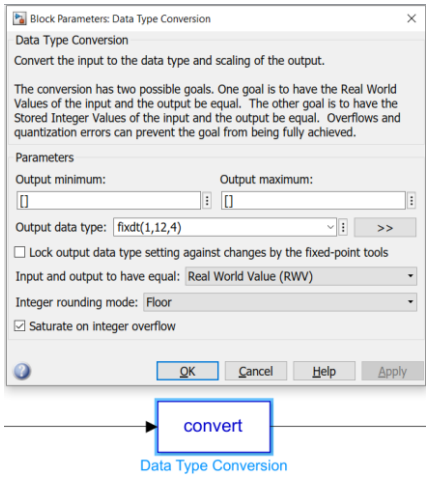

Figure 153 – Data converter in Simulink

Fig. 154 shows the fixed-point model of the system after inserting the converter blocks across all design components. This model was verified to ensure that functional correctness is preserved. The resulting Bit Error Rate (BER) remains low and acceptable ($0.064\ \%$), although

slightly higher than the floating-point model, due to the introduction of quantization noise during the fixed-point conversion process.

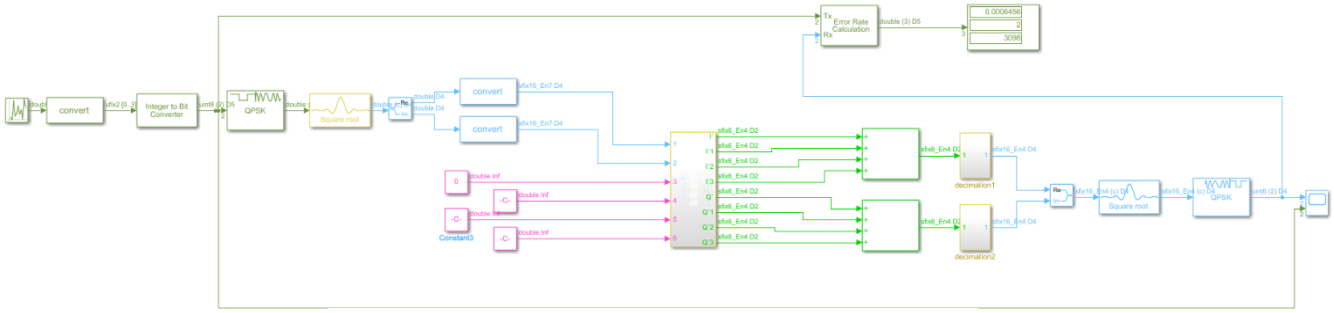

Figure 154 – Fixed point model in Simulink

5.1 RTL Implementation

This section contains the development of RTL code for each functional block, based on the verified fixed-point model, followed by integration into complete top-level modules for both the transmitter (Tx) and receiver (Rx) systems.

5.1.1 Clock divider

The system is driven by a single master clock, from which all other internal clocks are derived through clock division. This hierarchical clocking structure is essential to support the varying processing rates required by different blocks in the transceiver.

Receiver (Rx) Clocking:

- The ADC outputs an oversampled signal at the master clock rate.
- The Interleaver operates at clk/2, requiring a clock divided by 2.
- Subsequent blocks, including the Digital Down Converter (DDC) and Complex Weight Multiplication (CWM), also operate at clk/2.
- The Decimation filter includes:
    - A Comp filter stage that down samples the signal by 8, requiring a clk/16.
    - A Half-band filter that further down samples the signal by 2, requiring a clk/32.

Transmitter (Tx) Clocking:

The transmitter path performs the reverse operations, so corresponding clock domains (clk/2, clk/16, clk/32) are also required.

To generate these multiple clock signals from the master clock, a clock divider block is implemented. This block provides the necessary divided clock frequencies (clk/2, clk/16, clk/32) to synchronize the operation of each stage while maintaining data integrity across clock domains.

To generate the required lower-frequency clocks from the master clock, a clock divider is implemented using a simple binary counter.

This clock divider works by incrementing a 5-bit binary counter on every rising edge of the master clock. The individual output clocks are generated by tapping specific bits of the counter:

- clk_2 is taken from bit 0, toggling every clock cycle (effectively clk/2).
- clk_16 is taken from bit 3, toggling every 8 cycles (clk/16).
- clk_32 is taken from bit 4, toggling every 16 cycles (clk/32).

Each output provides a square wave at the corresponding divided frequency and is suitable for use in synchronizing lower-speed processing blocks.

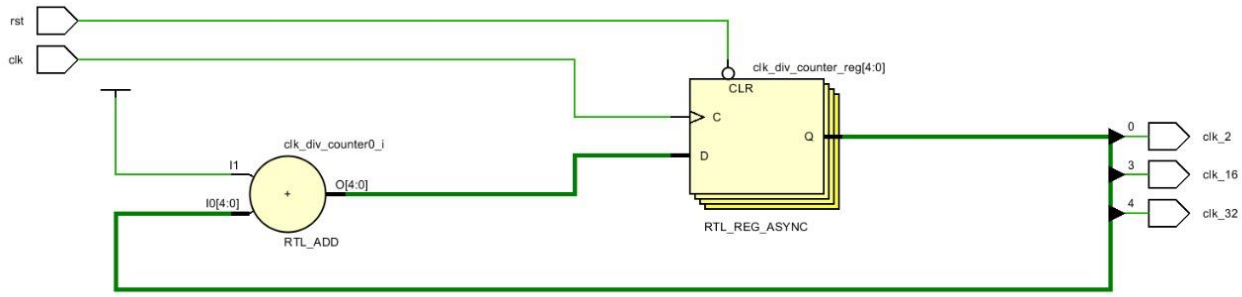

Figure 155 – Clock divider elaborated design

5.1.2 RTL implementation for Rx blocks

Interleaver

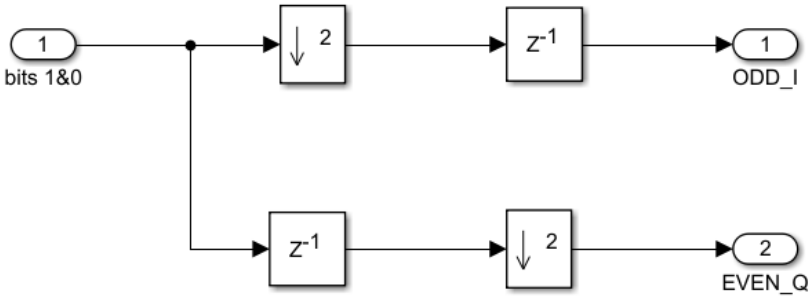

Figure 156 – Interleaver Simulink model

The interleaver takes a single serial data stream (labelled bits 1&0) and splits it into two separate sub-streams ODD_I (Odd-indexed bits for In-phase component) and EVEN_Q (Even-indexed bits for Quadrature component). The system receives a serial bitstream which is 2-bit output from the Delta-Sigma ADC operating at the sampling frequency fs, labeled "bits 1&0" that carries interleaved I and Q data **(**$x_0, x_1, x_2, x_3, x_4, x_5, \ldots$**).**

The upper branch contains a down sampler which takes every second sample from the input stream, since it's the first operation, it selects **(**x_0, x_2, x_4 …**)** even-indexed values. The z^{-1} stage introduces one-sample delay. Effectively shifts the selected even-indexed values by one, yielding (x_2, x_4, x_6 ….)**.** This aligns it to pick odd-indexed I values, due to the initial down sampling and delay, so the output values are (x_1, x_3, x_5 …) of the original stream.

The lower branch contains a z^{-1} stage that delays the input stream by one sample yielding (x_1, x_2, x_3 ….)**.** The down sampler takes every second sample from the input stream, since it's the second operation, it selects **(**x_1, x_3, x_5 …**)** odd-indexed values**.** This aligns it to pick even-indexed Q values, due to the initial delay and down sampling, so the output values are (x_0, x_2, x_4 …) of the original stream.

The two branches complement each other, working in parallel to extract an output with a rate of $F_s/2$ to be fed to the Digital Down Converter.

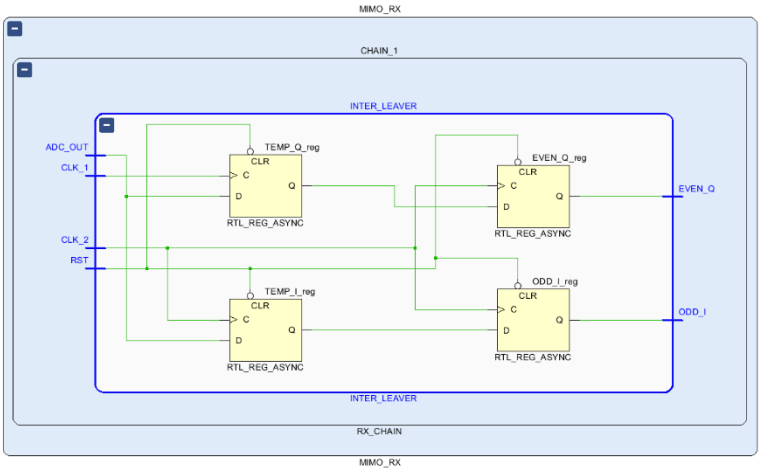

Figure 157 – Interleaver elaborated design

DDC

In traditional Digital Down Conversion (DDC), the incoming signal is multiplied by a Local Oscillator (LO) signal to shift it from the Intermediate Frequency (IF) to baseband. This LO is typically a sinusoidal wave, requiring costly multipliers and increased complexity.

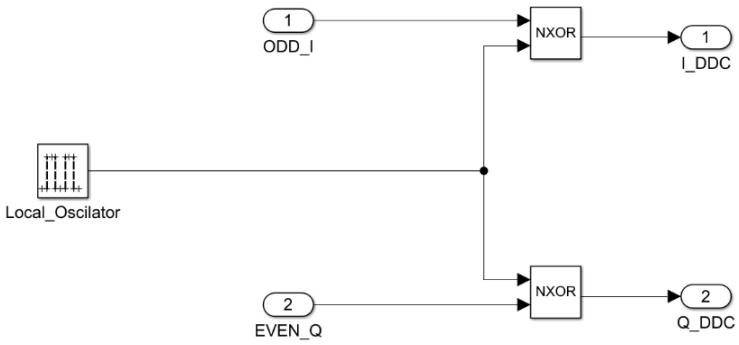

Figure 158 – DDC Simulink model

With $IF = f_s/4$ and the use of an Interleaver which removes samples that would be multiplied by zero in the LO waveform, only non-zero LO values: +1 and -1 are left, forming a square wave at $f_s/2$ which allows us to replace expensive multipliers with simple digital logic XNOR gate.

For the local oscillator (LO), sine and cosine waves are produced from a CORDIC block that will be explained shortly. The input signal is multiplied with a cosine wave to extract the In-phase (I) component and with a sine wave to extract the Quadrature (Q) component.

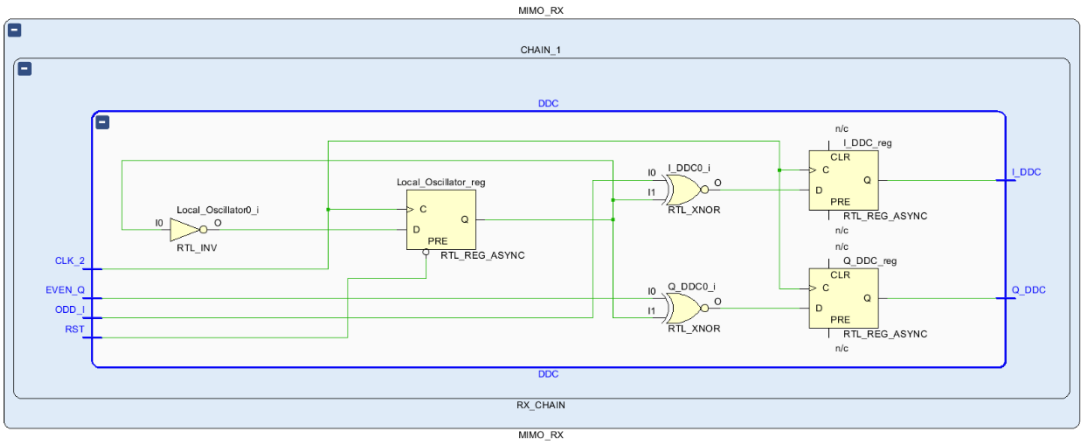

Figure 159 – DDC elaborated design

CWM

In digital beamforming, phase shifting is essential to steer the beam in the desired direction. This is implemented in each antenna path using Complex Weight Multiplication (CWM), applied to the In-phase (I) and Quadrature (Q) components.

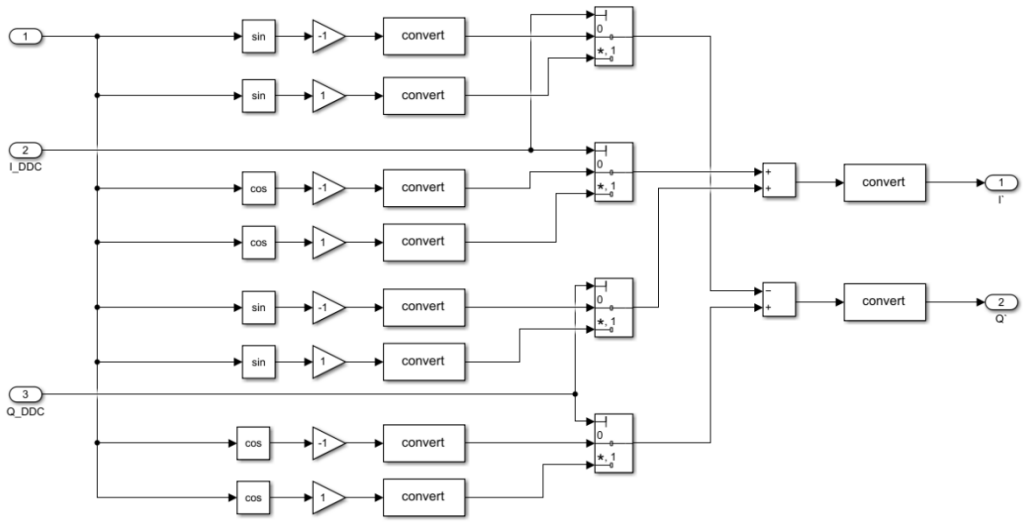

Figure 160 – RX CWM Simulink model

In the Bit-Stream Processing (BSP) architecture implemented within the receiver chain, the In-phase (I) and Quadrature (Q) signals are represented as single-bit data streams. Due to this single-bit representation, traditional multipliers are replaced by multiplexers (MUXs), which perform equivalent operations more efficiently in hardware. The sine and cosine waveforms required for complex weight multiplication are generated using a CORDIC.

To perform the final summation across all antenna elements or sub-chains, two adders are used, one adder for summing the rotated I' components and another for summing the Q' components.

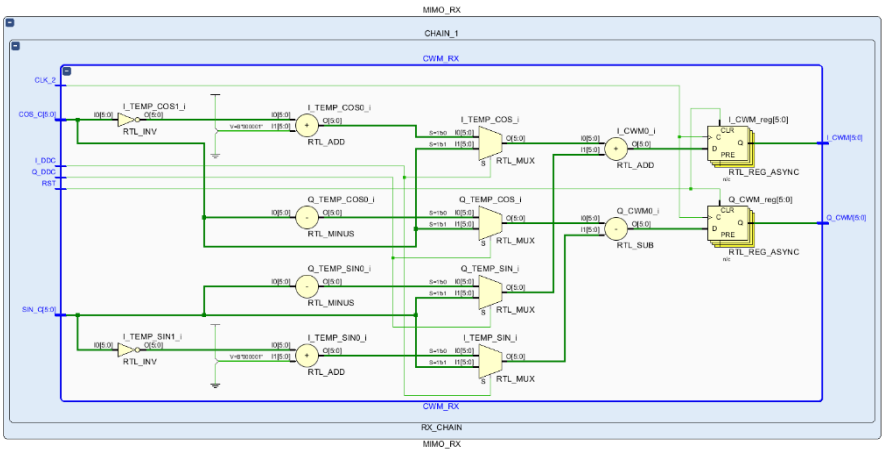

Figure 161 – RX CWM elaborated design

Decimation

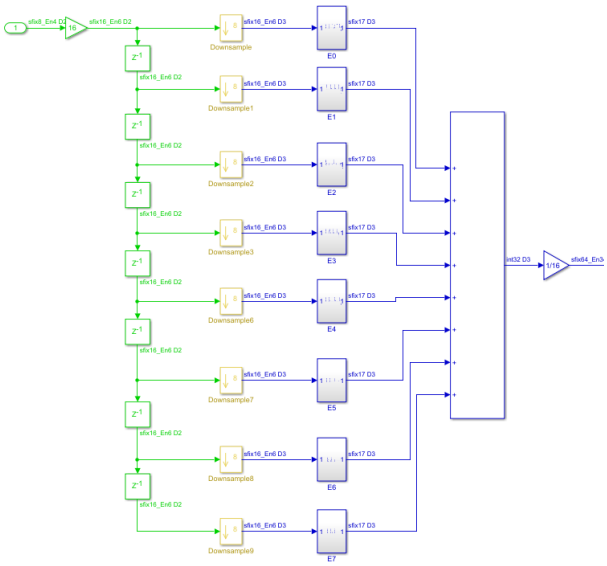

Figure 162 – Comb filter in the decimation

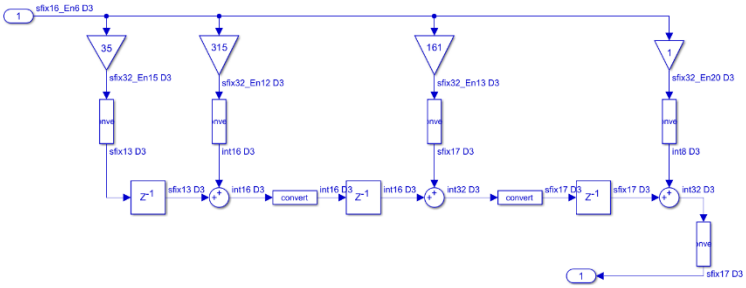

Figure 163 – E_0 in the comb filter

Fig. 162 shows the comb filter in the decimation process which consists of the following:

Input gain stage - An input gain stage scales fractional input values by 2^N (where N represents the fractional bit-width, typically N=4 for $gain = 16$),

converting them into fixed-point integers to preserve precision during processing and simplify hardware mapping.

Unit delay (z^{-1}) - Each unit delay is realized using a flip-flop, providing single-cycle storage with minimal area overhead.

Down sampler - A counter-based decimator reduces the sampling rate by an integer factor M using a modulo-M counter. When the counter resets to zero, the current input sample is captured; during subsequent (M−1) counts, incoming samples are systematically discarded.

Polyphase components - There are eight polyphase components in the comb and each one of them consists of constant gains and unit delays. Fig. 163 shows E_0 in the comb filter. The constant-coefficient gains are optimized during synthesis rather than using power-hungry multipliers, the tool automatically converts these fixed gains into combinations of shifters and adders.

Final Output Scaling and Bit Precision Recovery - Following the polyphase filtering and down sampling operations, the system performs output scaling to recover the original fractional precision.

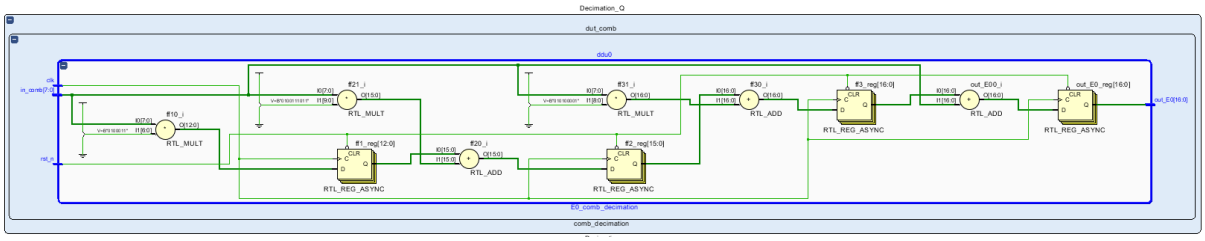

Figure 164 – Comb filter E_0 elaborated design

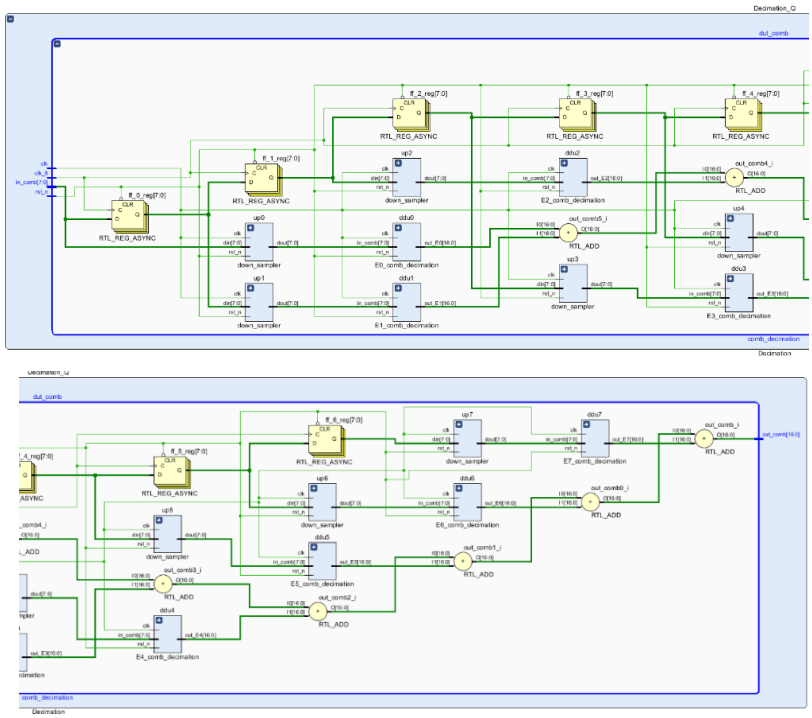

Figure 165 – Comb filter elaborated design

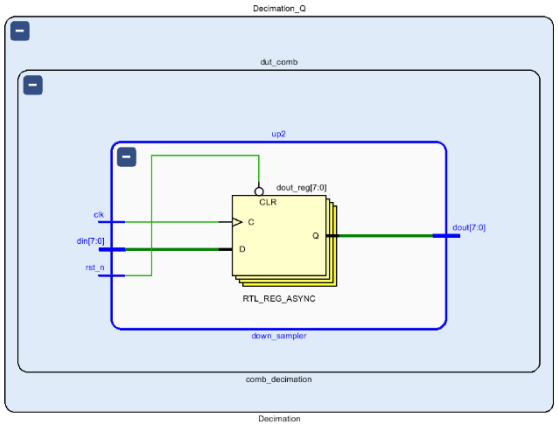

Figure 166 – Down sampler elaborated design

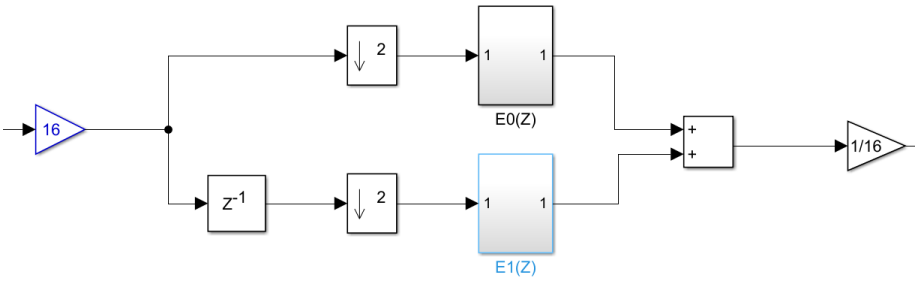

Figure 167 – Half band filter in the decimation

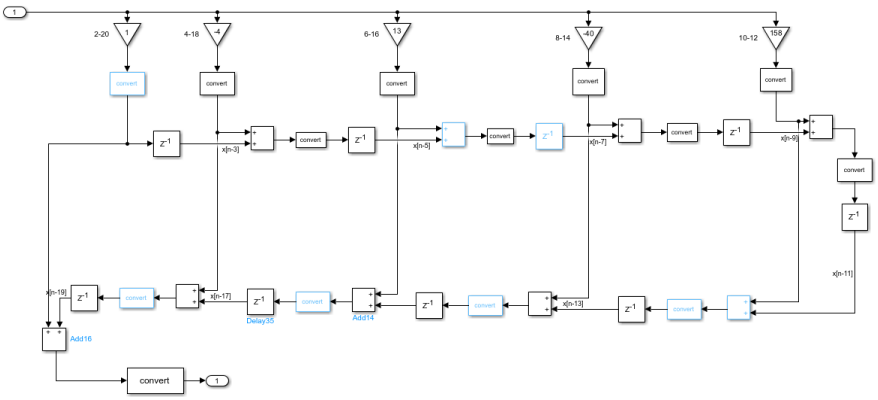

Figure 168 – E_1 in the half band filter

Fig. 167 shows the structure of the half-band filter, which is composed of the same fundamental building blocks as the comb filter. Due to this architectural similarity, the half-band filter can be mapped to RTL code using the same design methodology as the comb filter.

Polyphase components – In the polyphase filter implementation, a key distinction between the comb filters and the half-band filters lies in their coefficient characteristics. Comb filters utilize integer-valued coefficients, which naturally suit fixed-point digital hardware. In

contrast, half-band filters use fractional coefficients, requiring additional handling to be implemented efficiently in hardware.

To accommodate fixed-point RTL implementation, all fractional coefficients of the half-band filter are scaled by a factor of 512 (why 512? Because it is the first value that would make all the coefficients integer), as shown in Fig. 168. This scaling transforms the fractional values into integers, making them compatible with integer-based arithmetic units in RTL. However, to maintain the correct signal amplitude after filtering, a normalization step is necessary.

This normalization, shown in Fig. 167, is applied after the half-band stage and is implemented in RTL as a logical right shift operation. This shift effectively divides the signal by the same scaling factor (512), restoring the correct amplitude while avoiding the need for explicit division operations, which are resource-intensive in hardware

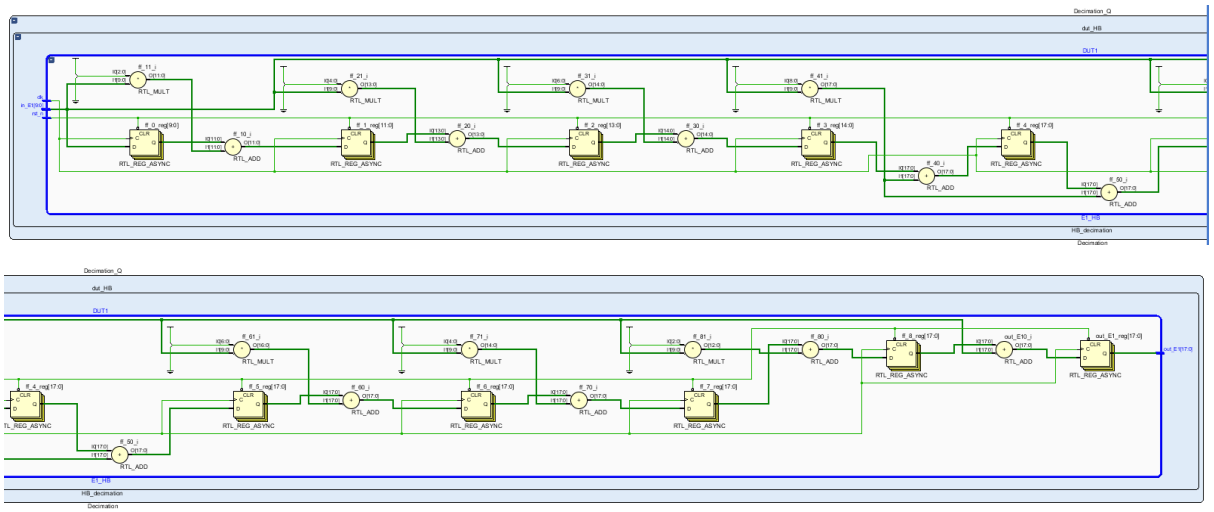

Figure 169 – Half band E_1 elaborated design

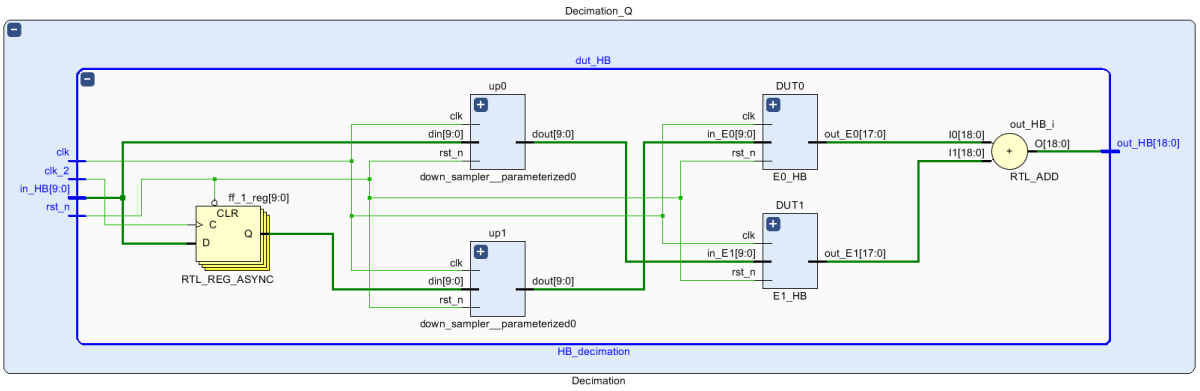

Figure 170 – Decimation half band filter elaborated design

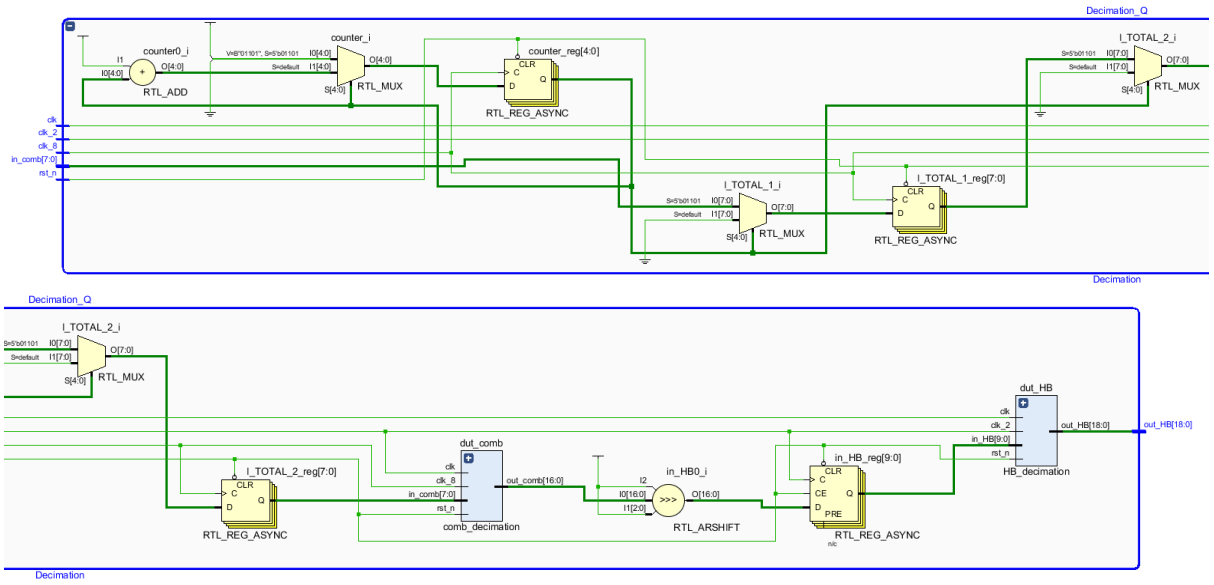

Figure 171 – decimation filter elaborated design

Rx integration

A top-level module was developed to integrate four independent receiver chains, each containing the full sequence of processing blocks required for signal reception. Clock dividers and CORDIC modules are also included.

This complete top-level structure was verified for functional correctness and prepared as input to the next steps of the digital implementation flow, including synthesis, DFT insertion, and place-and-route (PNR).

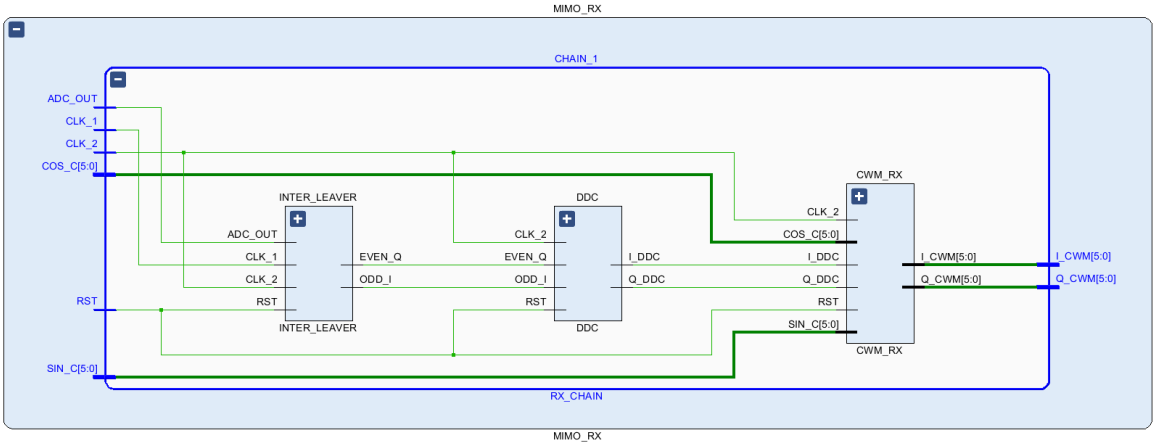

Figure 172 – Rx chain elaborated design

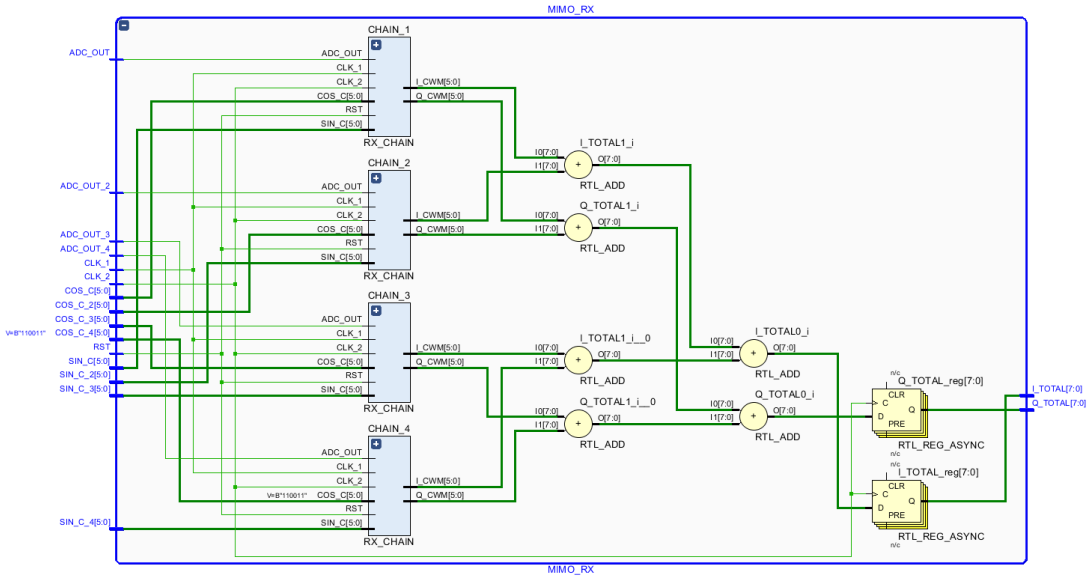

Figure 173 – 4 Rx chains Elaborated design

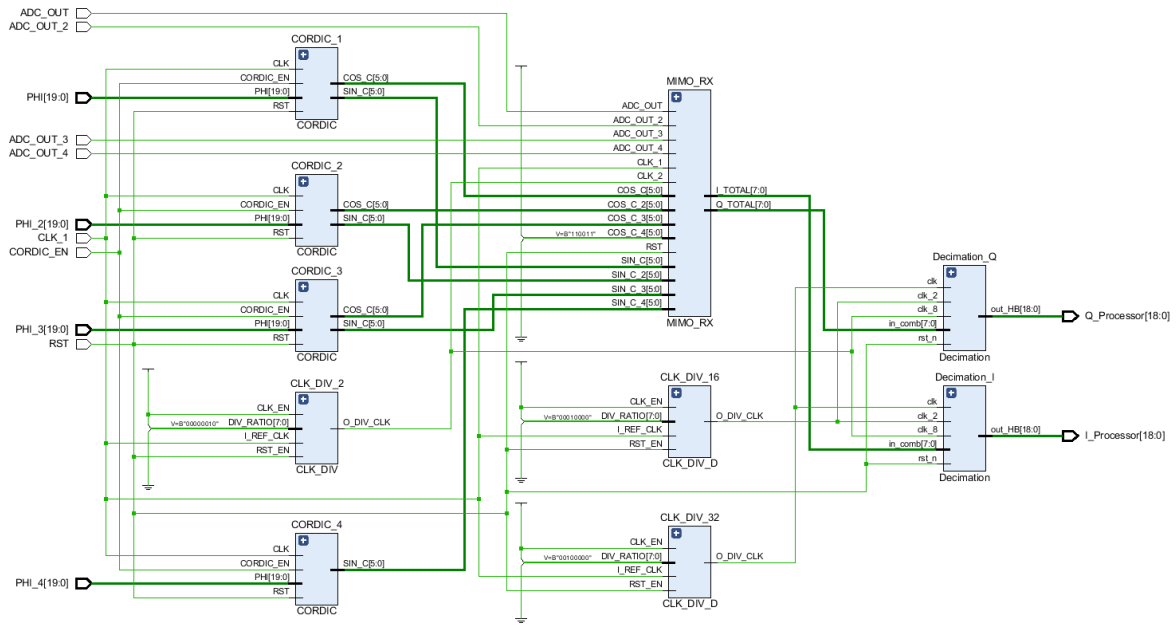

Figure 174 – MIMO Rx elaborated design

5.1.3 RTL implementation for Tx blocks

CWM

The CWM module implements a complex weighting and mixing of the input in-phase (I_) and quadrature components. This operation is essential in digital beamforming systems, enabling phase steering of transmitted signals by applying a controlled rotation using sine and cosine values. The module performs this rotation using fixed-point arithmetic, closely aligned with the Simulink fixed-point model.

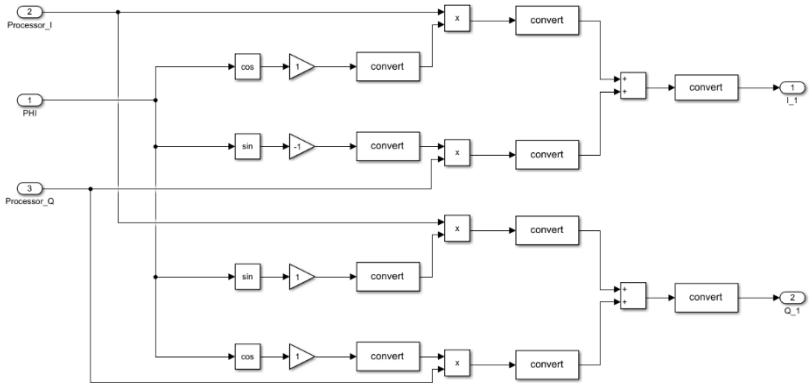

Figure 175 – CWM Simulink model

The inputs are I_processor, Q_processor (16-bit signed) Input baseband signals from processor and cos_ph, sin_ph (6-bit signed): Cosine and sine values of the phase rotation angle in fixed point format, from **CORDIC**.

The outputs are I_cwm, Q_cwm (18-bit signed) Phase-rotated output signals.

To match fixed-point behaviour in simulation, each product is computed in full 32-bit signed precision but bit truncation is applied to match fixed-point scaling.

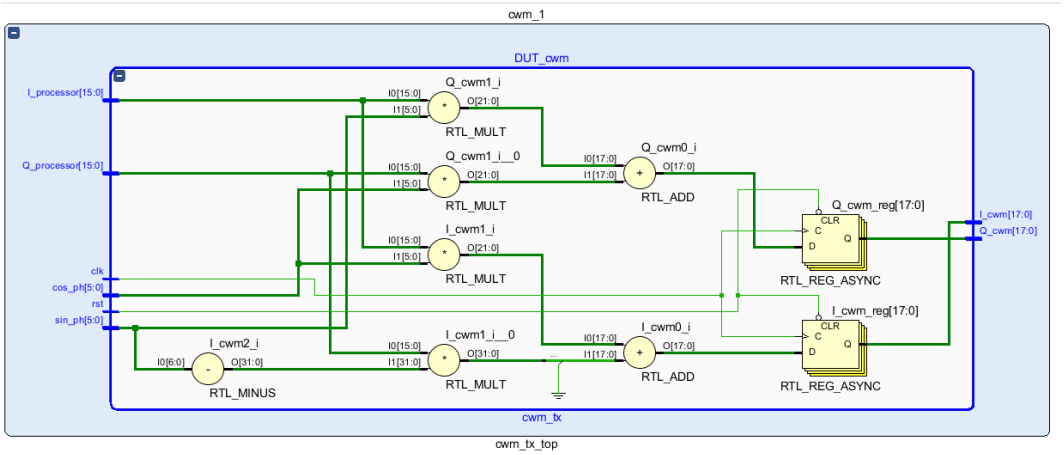

Figure 176 – Weight multiplication elaborated design

CORDIC

The module computes $cos(\theta)$ and $sin(\theta)$ for an input angle θ, using 16 iterations of the CORDIC rotation algorithm.

The input theta is a 16-bit signed fixed-point number (1 sign bit, 2 integer bits, 13 fractional bits). The outputs are 16-bit signed fixed-point numbers (1 sign bit, 1 integer bit, 14 fractional bits).

CORDIC starts with an initial vector:

$x_0 = K$ where $K \approx 0.607252935, \quad y_0 = 0, \quad z_0 = \theta$

The core algorithm performs 16 iterations of vector rotation, adjusting the angle towards zero, and updating x and y. After all iterations x is assigned to cos_theta and y is assigned to sin_theta.

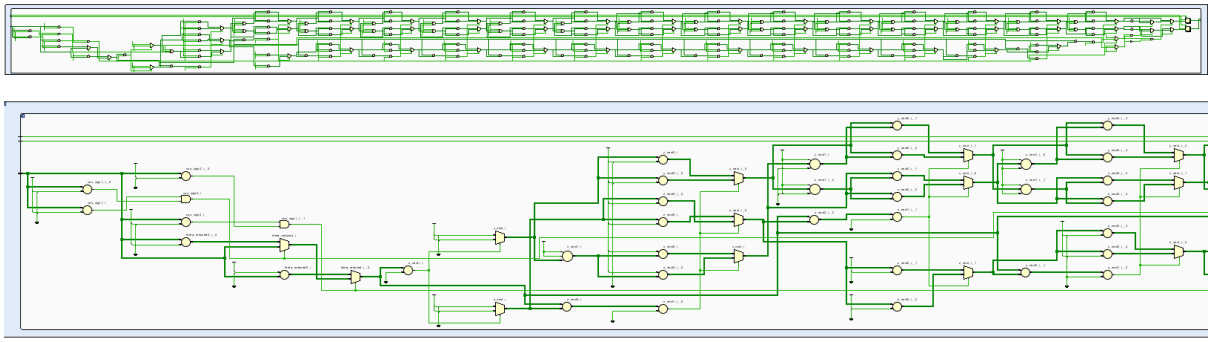

Figure 177 – CORDIC elaborated design

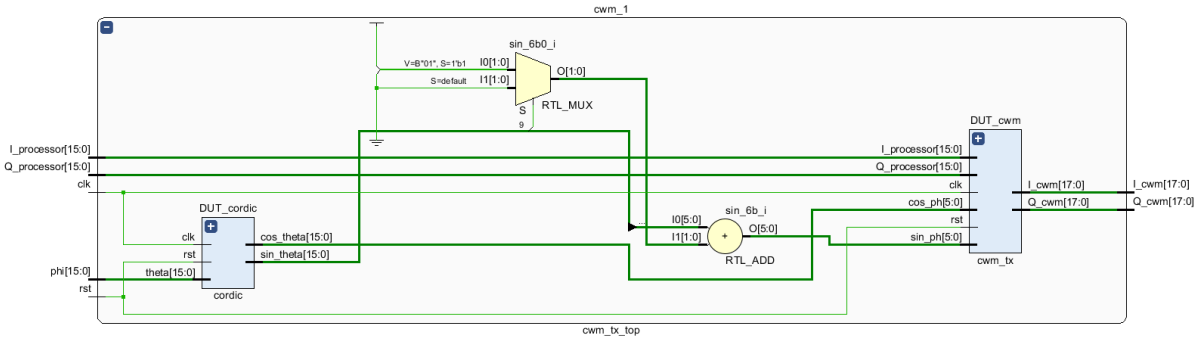

Figure 178 – Tx CWM elaborated design

Interpolation

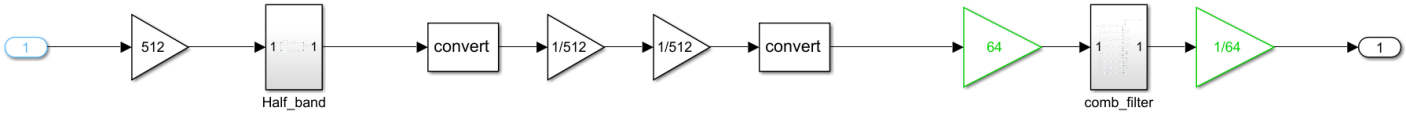

Figure 179 – Interpolation filter Simulink model

Fig. 179 shows interpolation filter Simulink model

The 512 gain – to scale fractional input values to integers (2^9 as 9 is the number of bits given to the fractional part for the input).

The first 1/512 gain – for the fractional coefficients as illustrated in decimation.

The second 1/512 gain – to recover the original fractional precision.

The 64 gain – also to scale fractional input values to integers.

The 1/64 gain – to recover the original fractional precision.

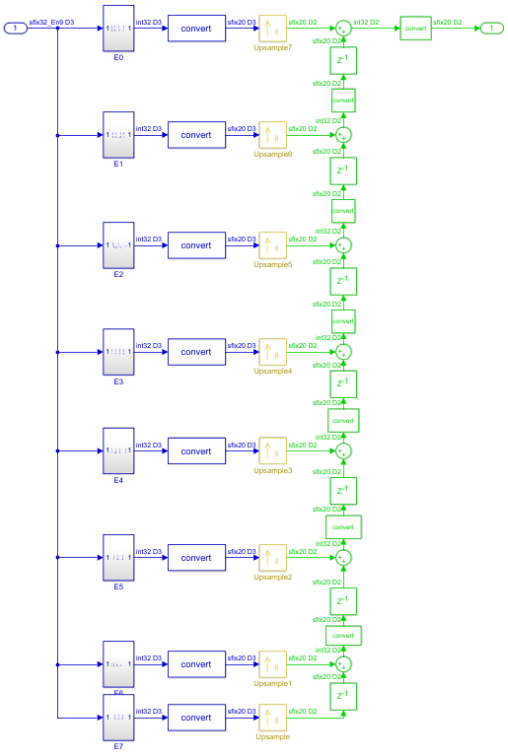

Figure 180 – Comb filter in interpolation

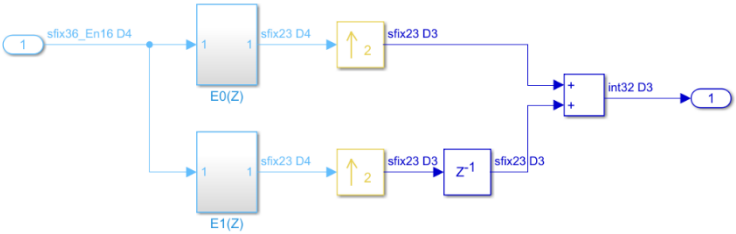

Figure 181 – Half band filter in interpolation

Both the comb filter and the half-band filter in the interpolation path shown in Fig. 179 and share the same fundamental building blocks described previously in the decimation process, such as adders, multipliers, and delay elements. However, a new functional block introduced in the interpolation chain is the up-sampling stage.

The up-sampling stage increases the sampling rate of the input signal by an integer factor M, and this is achieved through a two-step process:

1. **Clock Rate Increase**: The system transitions to a higher clock frequency equal to $M \times SFS$ (Sampling Frequency Scaling), effectively creating temporal space for the insertion of additional samples.

2. **Zero-Insertion (Zero Padding):** During this accelerated clock cycle, the system inserts $M - 1$ zero-valued samples between every original input sample. This process is commonly referred to as zero-padding and serves to up sample the signal in time without altering the original data values.

This zero-padded signal is then passed through the filters, which reconstruct the signal by smoothing the transitions between original and inserted zero samples. This process effectively increases the signal's sampling rate while maintaining signal integrity and minimizing aliasing artifacts.

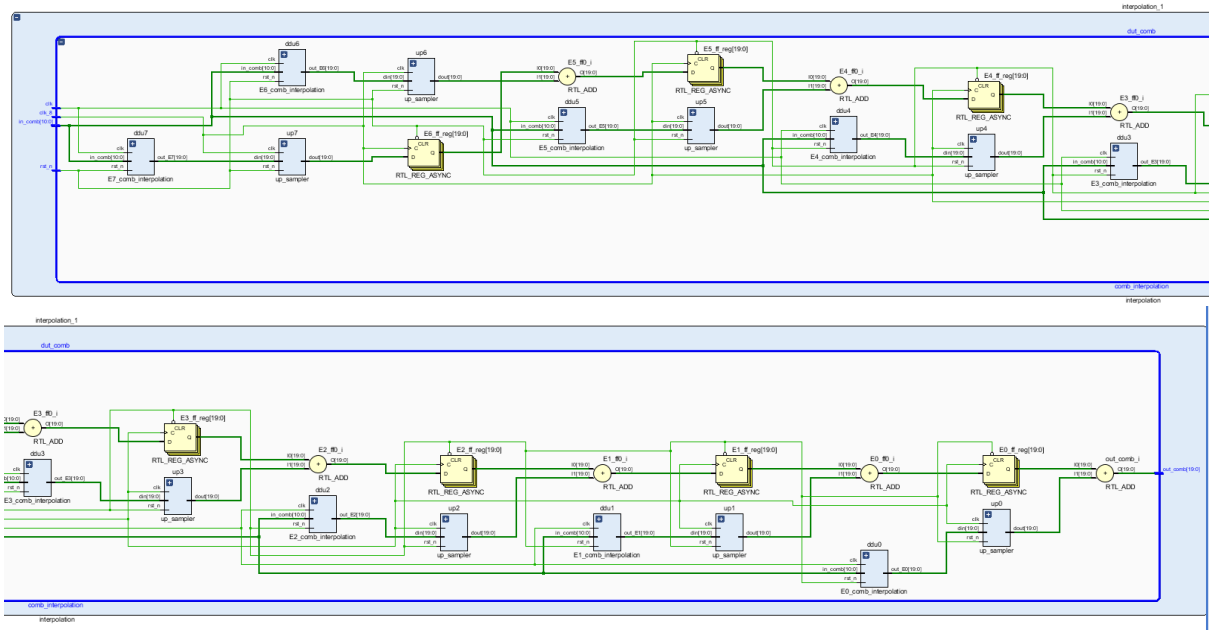

Figure 182 – Interpolation comb filter elaborated design

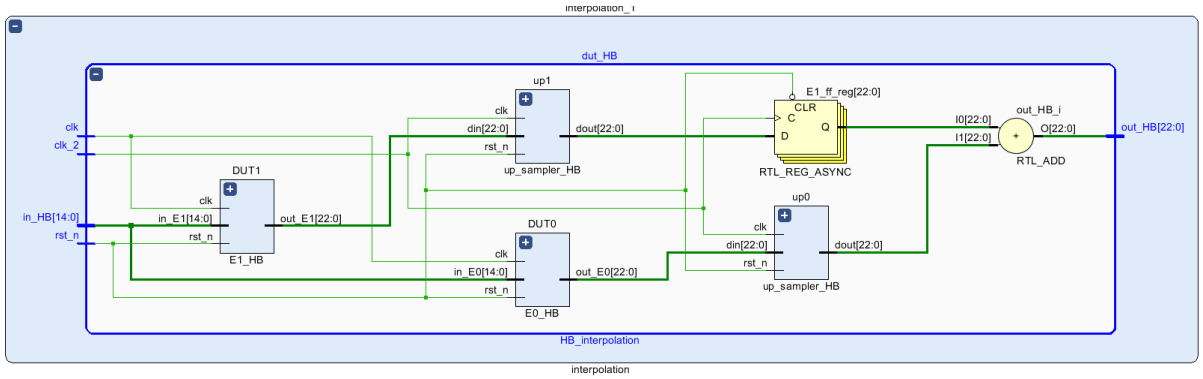

Figure 183 – Interpolation half band filter elaborated

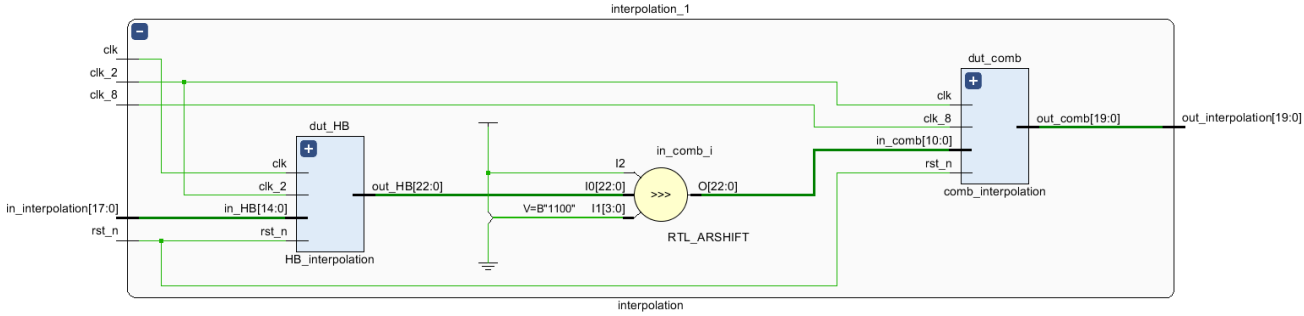

Figure 184 – Interpolation filter elaborated design

DDSM

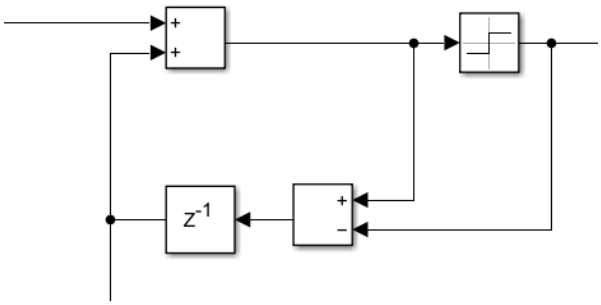

Figure 185 – A single stage of the MASH DDSM Architecture

Each processing stage in the Simulink model comprises four key components: two adders, one delay block, and one sign detection block. These are mapped to efficient RTL equivalents during hardware implementation.

- **Adders:** Both adders are implemented using standard full adder structures.
- **Delay Block:** The delay block is implemented as a single-register element. Its primary purpose is to introduce a one-cycle delay in the data path to support feedback within the system.
- **Sign Block:** The sign detection logic is realized using a 4-to-1 multiplexer (MUX). The select inputs to this MUX are derived as follows:
  - The first select bit is the most significant bit (MSB) of the input, which indicates the sign of the number in two's complement representation.
  - The second select bit is obtained via an OR-reduction of all remaining (non-MSB) bits, which serves to identify whether the input is zero or a non-zero value.

MUX selector bits	Output
00	0
01	1
10	-1
11	-1

This sign detection approach was chosen for its simplicity and effectiveness, and it ensures that zeros propagate through the system unaltered. Allowing a zero input to be interpreted incorrectly as either +1 or −1 would lead to unintended subtractions when the block has no valid input at startup, resulting in unpredictable behavior and an unknown accumulation of error over multiple cycles.

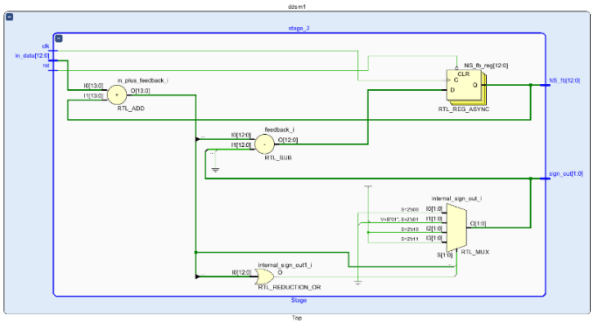

Figure 186 – Single stage MASH DDSM Architecture elaborated design

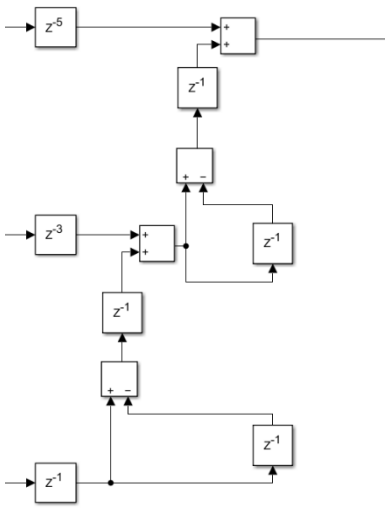

Figure 187 – Pipelined MASH Noise Cancellation network

The noise cancellation network is constructed using a chain of delay and addition elements. In the RTL implementation, these are realized as cascaded registers and full adders. The registers serve as programmable delay elements, enabling precise control over signal alignment, while the full adders perform accumulation across multiple paths.

This structure supports dynamic cancellation of quantization noise by replicating and delaying signal components across multiple branches. The resulting summation reinforces desired signal content while destructively interfering with noise components effectively shaping the spectral distribution of quantization artifacts.

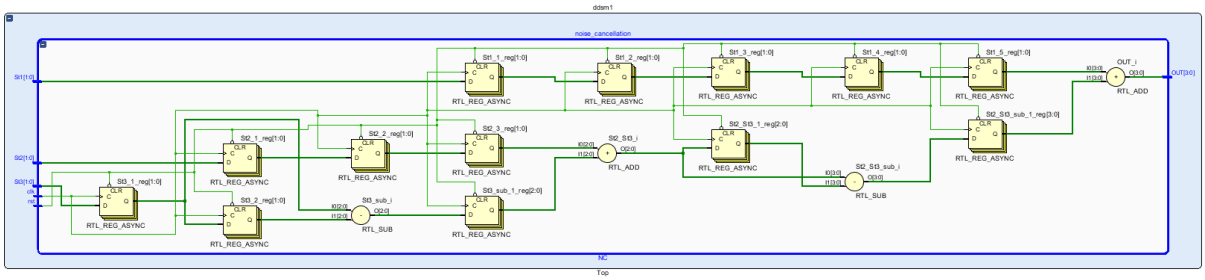

Figure 188 – Pipelined MASH Noise Cancellation network elaborated design

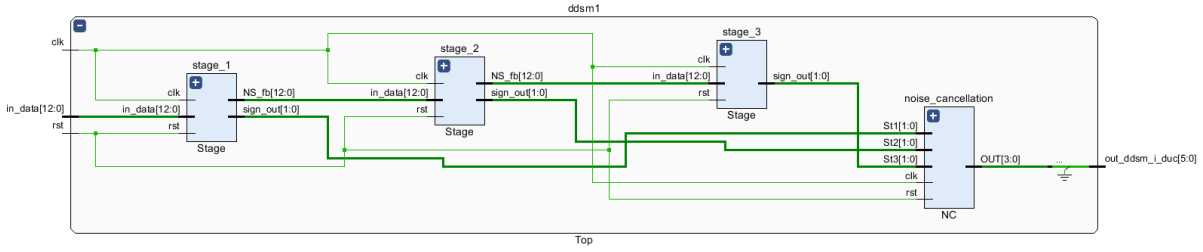

Figure 189 – DDSM elaborated design

DUC

The I_DUC and Q_DUC inputs are generated in a slower clk/2 clock domain, while the Digital Up-Converter (DUC) module operates in the faster clk domain. Inside the DUC, these inputs are mixed with internally generated sine and cosine waveforms, derived from a 4-phase counter, to perform digital modulation. Since all signal processing—including mixing and filtering—occurs at the higher clk frequency, the output OUT_DUC is valid in the clk domain.

Inputs are 6-bit signed baseband in-phase and quadrature inputs to be upconverted. The output OUT_DUC is 7-bit signed after up conversion and summation.

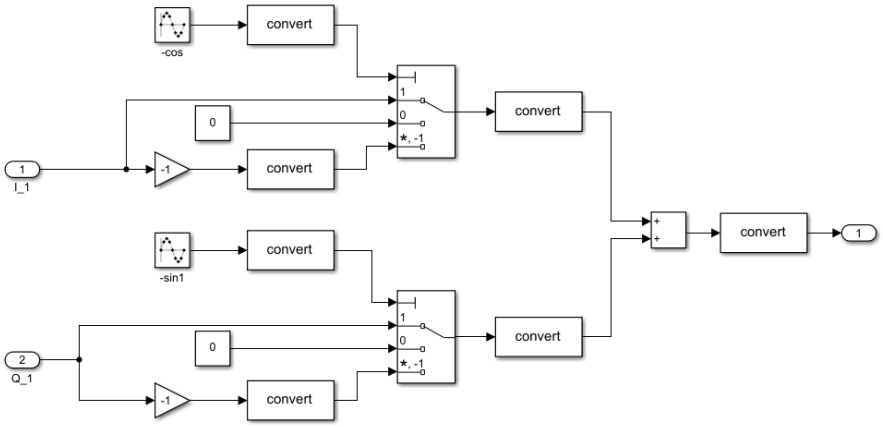

Figure 190 – DUC Simulink model

A 2-bit counter (i) that cycles through values 0 to 3 generates quadrature phase control. At each counter state, corresponding signed values are assigned to cosine and sine signals:

$$
\begin{aligned}
i &= 0\text{: } (sin = 1, \quad cos = 0) \\
i &= 1\text{: } (sin = 0, \quad cos = -1) \\
i &= 2\text{: } (sin = -1, \quad cos = 0) \\
i &= 3\text{: } (sin = 0, \quad cos = 1)
\end{aligned}
$$

Based on these control values, the input $I\ and\ Q$ components are conditionally multiplied by +1, -1, or 0 based on using $F_s = 4F_c$. This multiplication is implemented using conditional assignment rather than multipliers, improving hardware efficiency. After computing the rotated $I\ and\ Q$ components, the output `OUT_DUC` is generated by summing them.

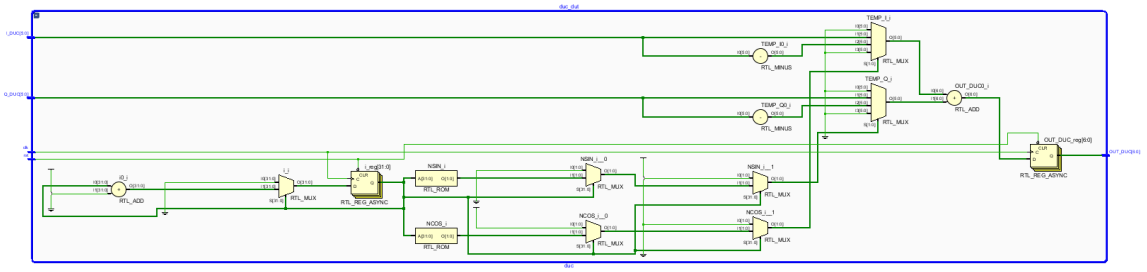

Figure 191 – DUC elaborated design

Tx integration

A top-level module was developed to integrate all functional blocks of the transmitter chain, including the Clock Divider. This unified module serves as the entry point for the subsequent stages of the digital design flow, such as DFT insertion, synthesis, formal verification, and place-and-route (PNR).

The top-level module ensures proper interconnection and synchronization between all sub-blocks and facilitates simulation and validation of the complete transmission path.

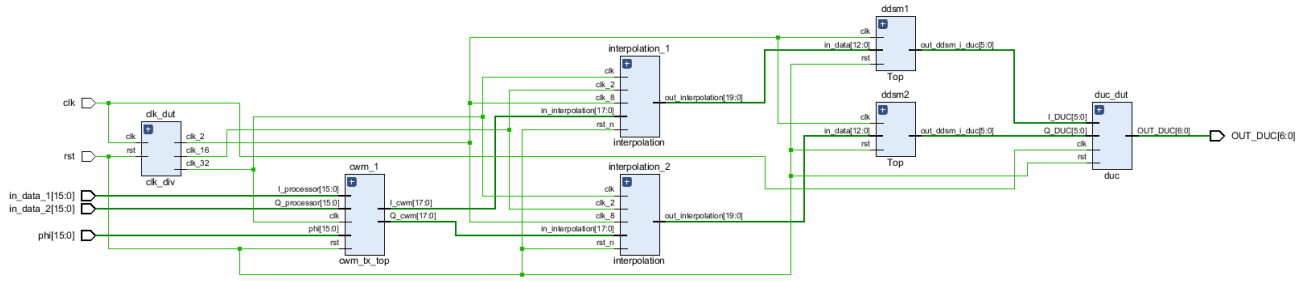

Figure 192 – Transmitter chain elaborated design

5.2 RTL verification

To verify the correctness of each individual block, a verification approach is employed. Specifically, the input and output signals from the Simulink model are extracted as shown in Fig. 193. The Simulink-generated input is then applied to the corresponding RTL implementation of the block. The output produced by the RTL is subsequently compared against the Simulink output. This bit-accurate comparison ensures functional equivalence between the high-level model and the hardware description, and helps identify any discrepancies arising from fixed-point quantization, rounding, or implementation-specific behavior.

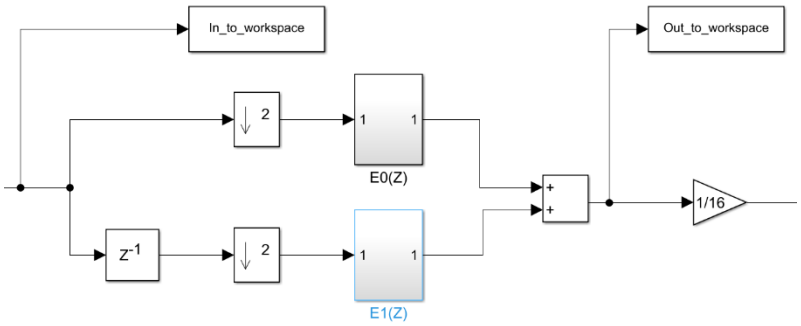

Figure 193 – Simulink In/Out extraction

Fig. 194 presents an example of the verification process for the decimation block. In the waveform, mem_out, shown at the top of the figure, represents the output generated by the RTL implementation. This output is compared against mem_check, shown at the bottom, which contains the reference output obtained from the Simulink model.

To detect mismatches between the two outputs, a comparison logic is implemented in the testbench. This logic continuously checks for bit-level equivalence between mem_out and mem_check. If a discrepancy is detected, a counter signal increments to indicate the number

of wrong outputs as shown in Fig. 195. This verification method ensures the RTL behavior matches the high-level design, confirming functional correctness of the decimation block.

The same approach is used for all the blocks in the Rx/Tx chain.

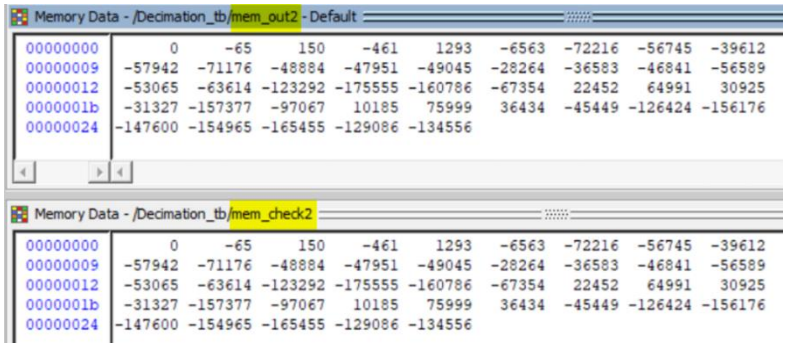

Figure 194 – decimation verification

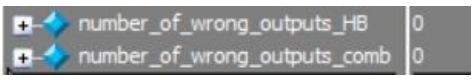

Figure 195 – number of wrong outputs

5.3 Logical Synthesis

Logic Synthesis is the process that translates, and maps RTL code written in HDL into a technology specific gate-level netlist, optimized for a set of pre-defined constraints.

Starting with a behavioural RTL design, standard cell library, a set of design constraints. And finishing with tech mapped gate-level netlist, Timing, Area and power reports. Cadence Genus is used for synthesis.

Standard cells used in digital design are characterized in the technology library under specific operating conditions, defined by Process, Voltage, and Temperature (PVT) corners. These PVT variations reflect the real-world manufacturing and environmental fluctuations that can affect the performance of digital circuits as seen in Fig. 196, so it must be considered during synthesis and timing analysis

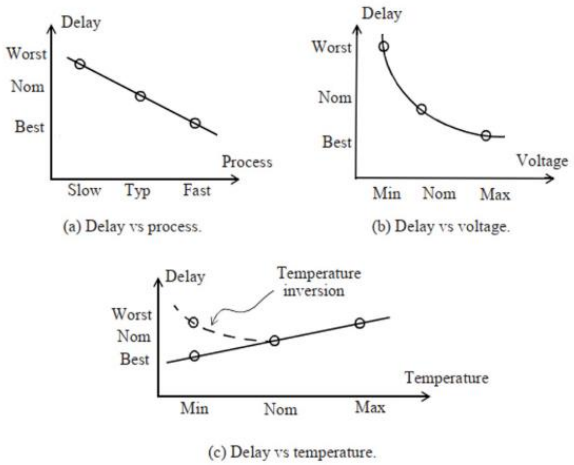

Figure 196 – PVT variations [196]

Two main TCL scripts were used to automate the digital synthesis process:

1- synthesis script

This script handles the full synthesis flow. It performs the following steps:

- Set up the environment: Define the top module name, working directory, RTL file's location and technology library path.
- Read design files: Load the RTL code and required technology libraries.
- Read constraints: Load timing and design constraints from a separate file (cons.tcl).
- Check design consistency: Ensure the design is logically correct before synthesis.
- Compile the design: Run synthesis, including mapping and optimization.
- Generate outputs:
    - Gate-level netlist: A synthesized version of the RTL using technology cells.
    - SDC file: Defines clocks and timing constraints for tools like synthesis and place-and-route.
    - SDF file: Provides detailed timing delays for simulation and timing verification.
    - Reports: Include area, power, and timing analysis summaries.

2- Constraints script

This file contains the design timing constraints and environment setup:

- Clock definitions: Define main and generated clocks.
- Clock relationships: Specify how clocks interact.
- I/O constraints: Set input/output delays and interface requirements.
- Driver and load settings: Assign driving cells and output loading.
- Operating conditions: Define temperature, voltage, and other conditions for accurate timing analysis.

Master clock	1 GHz
Generated clocks	Clk/2, clk/16, clk/32
Input delay	20% of the master clock
Output delay	20% of the master clock
Driving cells	Technology buffer used
Output load	0.1 pf
Operating conditions	Best and worst cases

Table 1 – Synthesis constraints

Results after synthesis for both Tx and Rx chains

Summarized synthesis results for Tx chain	
Frequency	$1\ GHz$
Total power	$6.9831\ mW$
Total area	$45537.48\ \mu m2$
Slack for the worst path	$526\ ps$

Table 2 – Tx synthesis results

Summarized synthesis results for 4 Rx chains	
Frequency	$1\ GHz$
Total power	$2.2415\ mW$
Total area	$83400.12\ \mu m2$
Slack for the worst path	$546\ ps$

Table 3 – Rx synthesis results

5.4 Formal Verification (Post-Synthesis)

LEC is also known as formal verification. Verification through simulation applies countless vectors to the circuit and afterward thinks about the subsequent output vectors to anticipated values. As designs become bigger and increasingly mindboggling and require more simulation vectors, regression testing with customary simulation tools turns into a bottleneck in the design flow. Formal verification utilizes mathematical methods to contrast the logic with being checked against either a logical detail or a reference structure. In contrast to verification through reproduction, formal verification does not require input vectors. As formal verification thinks about just logical functions amid correlations, it is autonomous of the plan's physical properties, for example, layout and timing. [40]

Cadence conformal LEC (Logical Equivalence Checking) was used for the formal verification. A “do file” which is automatically generated from synthesis is used in the equivalence checking. This script automates and streamlines the formal verification process using LEC to check the equivalence between the golden and revised models.

```
// Command: puts "No of compare points = [get_compare_points -count]"
// Command: get_compare_points -count
No of compare points = 2441
// Command: puts "No of diff points    = [get_compare_points -NONequivalent -count]"
// Command: get_compare_points -NONequivalent -count
No of diff points    = 0
// Command: puts "No of abort points   = [get_compare_points -abort -count]"
// Command: get_compare_points -abort -count
No of abort points   = 0
// Command: puts "No of unknown points = [get_compare_points -unknown -count]"
// Command: get_compare_points -unknown -count
No of unknown points = 0
```

Figure 197 – Tx LEC compare results

```
// Command: puts "No of compare points = [get_compare_points -count]"
// Command: get_compare_points -count
No of compare points = 5934
// Command: puts "No of diff points    = [get_compare_points -NONequivalent -count]"
// Command: get_compare_points -NONequivalent -count
No of diff points    = 0
// Command: puts "No of abort points   = [get_compare_points -abort -count]"
// Command: get_compare_points -abort -count
No of abort points   = 0
// Command: puts "No of unknown points = [get_compare_points -unknown -count]"
// Command: get_compare_points -unknown -count
No of unknown points = 0
```

Figure 198 – Rx LEC compare results

5.5 Design for Testability (DFT)

It is a design methodology aimed at enhancing the **observability** and **controllability** of internal signals within a digital circuit. By incorporating specific hardware structures, DFT facilitates the efficient detection of **manufacturing defects** and **logic faults** during post-silicon testing.

The scan chain technique which is one of the most commonly used DFT methods was implemented. This approach connects selected internal **flip-flops** into **serial shift-register chains**, enabling test patterns to be shifted in and the resulting states to be shifted out through a small number of I/O pins. This significantly improves test coverage while minimizing pin overhead.

The **Cadence Genus** tool was used to perform DFT insertion and validation.

1- RTL Modification

The initial step in the DFT process was to **modify the top-level RTL module** to enable **scan chain insertion**. This was done by introducing **multiplexers (MUXs)** in the clock path of sequential elements. Each MUX allows selection between the **functional clock** and the **scan clock**, based on the value of a dedicated **test mode control signal**.

- When **test mode = 0**:
  The design operates under normal functional conditions using the functional clock.
- When **test mode = 1**:
  The scan clock is enabled, allowing **scan shifting** and application of **ATPG-generated patterns** for testing.

This modification ensures that the design is compatible with scan-based testing while maintaining normal functionality during regular operation.

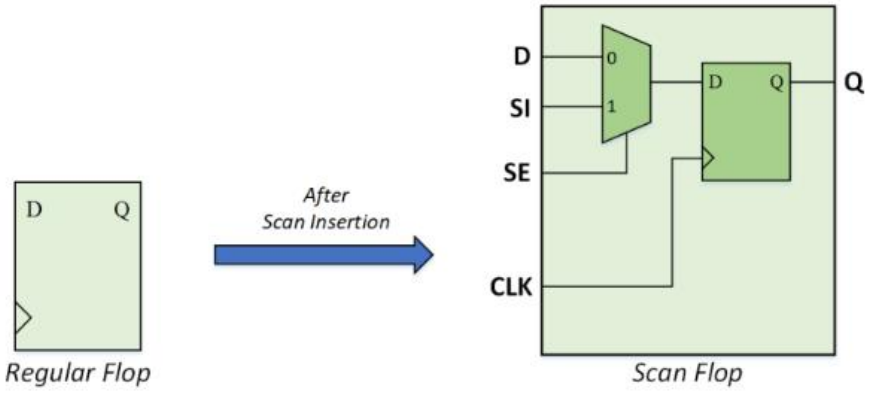

Figure 199 – Scan Flip Flop [41]

2- Synthesis script Modification

The synthesis script was modified to incorporate **Design for Testability (DFT)** features, ensuring that the design is fully testable post-fabrication. The following key actions were performed during this stage:

- **Definition of test control signals**, including:

    - test_mode
    - scan_enable
    - scan_in and scan_out ports
- **Conversion of all flip-flops** to **scan flip-flops** using equivalent standard-cell replacements from the technology library.
- **Automatic scan chain insertion**, ensuring correct **chaining and connectivity** of scan paths, while conforming to DFT design rules and minimizing routing overhead.

Generated Outputs

Upon successful DFT-aware synthesis, the following output files were produced:

- **Gate-Level Netlist**:
  A Verilog netlist including the inserted **scan chains**.
- **SDC File (Synopsys Design Constraints)**:
  Contains updated **timing constraints** reflecting scan-related paths and test mode conditions.
- **SDF File (Standard Delay Format)**:
  Used for **gate-level simulations** with post-synthesis delays.
- **Reports**:
  Comprehensive reports including:
    - **Area and power estimates**
    - **DFT rule checks**
    - **Timing analysis summaries**

These outputs form the foundation for downstream formal verification, DFT validation, and place-and-route (PNR) processes.

3- Constraints script Modification

As part of the DFT integration, additional **timing constraints and setup configurations** were applied to accurately model scan-related behavior during test mode. These included the following steps:

- **Generation of a dedicated scan clock** specifically for shift operations during scan testing.
- **Definition of input delays** for test-specific control ports such as test_mode, scan_enable, and scan_in, ensuring realistic timing modeling during Automatic Test Pattern Generation (ATPG) and scan simulations.
- **Definition of output delays** for scan_out ports to emulate the timing behavior observed at the chip boundary during scan-based testing.
- **Assignment of appropriate driving cells** for test_mode, scan_enable, and scan_in ports to correctly model their electrical behavior and switching characteristics.

- **Specification of output load constraints** on the scan_out ports to reflect the expected downstream load and enable accurate timing analysis.
- **Activation of test mode** during synthesis and timing analysis by setting both test_mode and scan_enable signals to logic **'1'**. This ensures scan paths are properly considered during optimization.
    - During **normal functional operation**, these signals are set to **logic '0'**, disabling the scan logic and allowing the system to operate in its standard mode without any interference from test circuitry.

These constraints ensure that the design meets **timing closure requirements** under both functional and test conditions and that scan operations can be performed reliably after fabrication.

Results after DFT for both Tx and Rx chains

Summarized synthesis results for Tx chain	
Frequency	$1\ GHz$
Total power	$9.26\ mW$
Total area	$55202.4\ \mu m2$
Slack for the worst path	$515\ ps$

Table 4 – Tx DFT results

Summarized synthesis results for 4 Rx chains	
Frequency	$1\ GHz$
Total power	$3.141\ mW$
Total area	$104368.32\ \mu m2$
Slack for the worst path	$450\ ps$

Table 5 – Rx DFT results

DFT violations report

```
Total 0 violations (muxed_scan)

Async. set/reset Rule Violations
================================
Reporting 0 async set/reset violations

Clock Rule Violations
=====================
Reporting 0 clock violations

Abstract Segment Test Mode Violations
=====================================
Reporting 0 abstract segment violations

Shift Register Segment Violations
=================================
Reporting 0 shift register segment violations

Tristate net contention Violations
==================================
Reporting 0 tristate net contention violations

Potential Race Condition Violations
===================================
Reporting 0 potential race condition violations

X-source Violations
==================
Reporting 0 x-source violations

X-source Violations By Instance
===============================
Reporting 0 instances generating observable x-sources
```

Figure 200 – DFT violations report

Coverage report generated

```
    Uncollapsed Stuck Fault Summary Report
-----------------------------------------------
fault class                     code   #faults
------------------------------  ----  ---------
Detected                         DT     129707
Possibly detected                PT          0
Undetectable                     UD       1824
ATPG untestable                  AU        163
Not detected                     ND       1634
-----------------------------------------------
total faults                             133328
test coverage                             98.63%
-----------------------------------------------
```

Figure 201 – Tx coverage report

```
    Uncollapsed Stuck Fault Summary Report
-----------------------------------------------
fault class                     code   #faults
------------------------------  ----  ---------
Detected                         DT     204933
Possibly detected                PT          0
Undetectable                     UD       2731
ATPG untestable                  AU         98
Not detected                     ND         48
-----------------------------------------------
total faults                             207810
test coverage                             99.93%
-----------------------------------------------
```

Figure 202 – Rx coverage report

5.6 Formal Verification (Post-DFT)

```
// Command: puts "No of compare points = [get_compare_points -count]"
// Command: get_compare_points -count
No of compare points = 2795
// Command: puts "No of diff points    = [get_compare_points -NONequivalent -count]"
// Command: get_compare_points -NONequivalent -count
No of diff points    = 0
// Command: puts "No of abort points   = [get_compare_points -abort -count]"
// Command: get_compare_points -abort -count
No of abort points   = 0
// Command: puts "No of unknown points = [get_compare_points -unknown -count]"
// Command: get_compare_points -unknown -count
No of unknown points = 0
```

Figure 203 – Tx LEC compare results

```
// Command: puts "No of compare points = [get_compare_points -count]"
// Command: get_compare_points -count
No of compare points = 6006
// Command: puts "No of diff points    = [get_compare_points -NONequivalent -count]"
// Command: get_compare_points -NONequivalent -count
No of diff points    = 0
// Command: puts "No of abort points   = [get_compare_points -abort -count]"
// Command: get_compare_points -abort -count
No of abort points   = 0
// Command: puts "No of unknown points = [get_compare_points -unknown -count]"
// Command: get_compare_points -unknown -count
No of unknown points = 0
```

Figure 204 – Rx LEC compare results

5.7 Place and Route (PNR)

A fully functional RTL which is synthesized undergoes a design partition or physical hierarchy which is further taken to block level physical design and synthesis activity. This phase of ASIC flow will decide the number of chips per wafer and ultimately the cost. Initial power estimation will help the die design and also help choose the correct package. The physical hierarchy of the blocks undergoes block-level placement, routing, and static timing analysis (STA). This is used to decide the floor plan and the power plan, Clock tree synthesis, and placement of standard cells and routing them. [42]

To perform this step, the PVT corners that will be used are chosen. Generally, there are three main corners that need to be run, the slowest or worst case, the nominal case and the fastest or best case. With each corner different issues arise as their timing difference can provoke setup and hold violations. During the physical design, physical cells like boundary, filler and well tap cells are inserted. Boundary cells are dummy cells placed on the design edges with the only purpose to avoid damage on the standard cells placed near the edge during the manufacturing process. [43]

The physical implementation flow begins by importing a set of essential **input files** that define the design's logical, physical, and timing characteristics. These files include:

1. **Netlist File (Verilog)**:
   Contains the synthesized RTL logic of the design, describing the interconnection of standard cells.
2. **LEF Files (Library Exchange Format)**:
   - **Technology LEF (6m1t0f)**: Defines the physical rules, layers, and design rules for routing (6 metal layers, 1 threshold voltage, 0 fins).
   - **Standard Cell LEF**: Describes the physical dimensions, pin locations, and routing blockages of the standard cell library.
3. **Capacitance Table (Cap Table – 6metal cap)**:
   Provides parasitic extraction information for the 6-metal layer process. It is used to accurately model resistance and capacitance values during RC extraction and timing analysis.
4. **.scandef File**:
   Specifies the placement of scan-related elements such as scan cells and chains. It is used during scan chain reordering and DFT implementation.
5. **.sdc Files (Synopsys Design Constraints)**:
   - Two separate **SDC files** may be provided (e.g., from the **DFT tool**), defining timing constraints, clock definitions, input/output delays, and false paths for synthesis and placement stages.
6. **.lib Files (Liberty Format)**:
   Provide **timing, power, and functional models** of standard cells and memory elements, used during timing analysis and power estimation.

These input files are essential for guiding the **place-and-route** flow, ensuring proper placement, routing, timing closure, and ultimately a successful design sign-off.

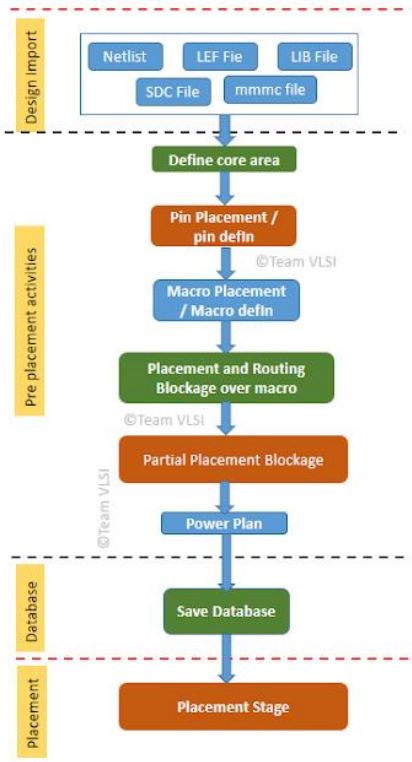

Figure 205 – Pre placement activities in PNR [45]

1- Floor planning

The goal of the floor planning stage is to create a layout that is routable and supports timing closure. The chip dimensions are defined with a square aspect ratio (H/W = 1), core utilization of 70%, and 60 μm margins on all sides between the core and die boundary.

Key aspects of floor planning include:

- **Chip Size and Aspect Ratio**: A square layout is chosen to simplify placement and routing.
- **Core Area Definition**: The core can be resized in both X and Y directions by up to 5% to optimize area and routing.
- **Core Utilization**: Set at 70% to allow for buffers and routing resources.
- **Core-to-IO Spacing**: Reserved space allows for power rings and avoids congestion near chip edges.

During pin placement, I/O pins are distributed along all four edges of the chip. **Metal 3** is used for horizontal pins, and **Metal 4** for vertical pins, ensuring alignment with routing directions and design rules.

Finally, checks are performed to ensure all pins are legal and correctly connected, and that no partition pins are floating or missing.

2- power planning

A power plan is a very robust power grid structure to deliver power to all macros and standard cells available in the design without much IR drop in the power grid. power grid takes power from bumps on the top metal layer and it delivers power to the lowest metal layer in which standard cells follow pin available.
From bumps, power goes to power stripe and power stripe delivers power to the VDD and VSS rails. Macros get power directly from power stripe as in place of macro there are no power rails drawn. [45]

At the beginning of this step, the **global nets** are connected to **VDD** and **VSS** to establish the power distribution network.
The power mesh consists of three main components:

- **Rings**: Surround the chip and deliver VDD and VSS to the I/O pads. These are typically implemented using higher metal layers such as **Metal 5** and **Metal 6**.
- **Stripes**: Extend from the power rings into the core area, delivering power to internal logic. Stripes are placed using two orthogonal metal layers to minimize **IR drop**, with **Metal 6** used as the highest vertical layer.
- **Rails**: Provide local VDD and VSS connections to the standard cells. These are formed using **Metal 1** and **Metal 6** through a process known as **special routing**.

To reduce **static IR drop**, two common techniques are used:

- Increasing the **width** of power stripes.
- Increasing the **number** of power wires across the core.

Finally, a **geometry verification** step is performed to ensure layout correctness. The command verify_drc is executed to identify and resolve any **Design Rule Check (DRC)** violations.

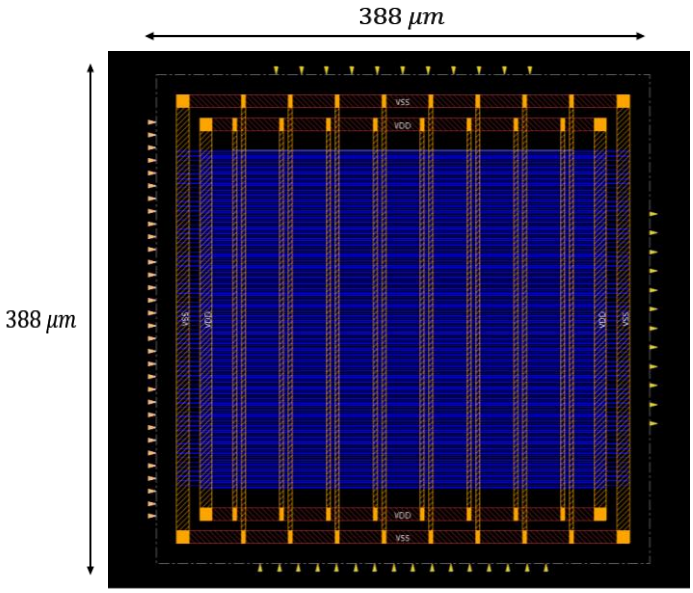

Figure 206 – Tx Power planning

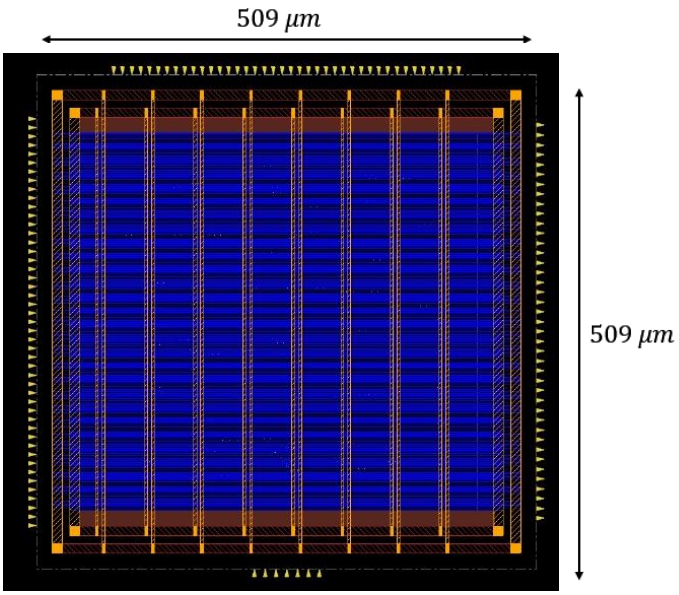

Figure 207 – Rx Power planning

3- Placement

The **placement stage** focuses on positioning the **standard cells** within the defined core area in an **optimal and legal** manner. The process is divided into two main phases:

1. **Global Placement**:
   Cells are distributed into regions (bins) to minimize wirelength and improve timing. At this stage, the tool does not enforce legality or prevent overlaps between cells.
2. **Detailed Placement (Legalization)**:
   Cells are adjusted and snapped to legal sites within the core boundary to ensure no overlap and adherence to design rules. The tool ensures optimal placement while considering timing, congestion, and area.

Placement Objectives

- Ensure **legal and optimized placement** of standard cells.
- Minimize **routing congestion** and maintain optimal placement density.
- Enhance **timing, area, and power** by optimizing high fanout nets and buffer insertion.

Design Placement Steps

1. **Load. SCAN DEF** file containing scan-related placement information.
2. **Add placement blockages** around the core to prevent routing congestion near I/O ports.
3. **Place standard cells** within the core area.
4. **Insert Tie-Low and Tie-High cells** to stabilize undefined logic levels and prevent voltage-induced glitches.
5. **Reorder scan chains** to optimize scan chain paths.
6. Perform final **placement optimization** to improve timing and layout quality.

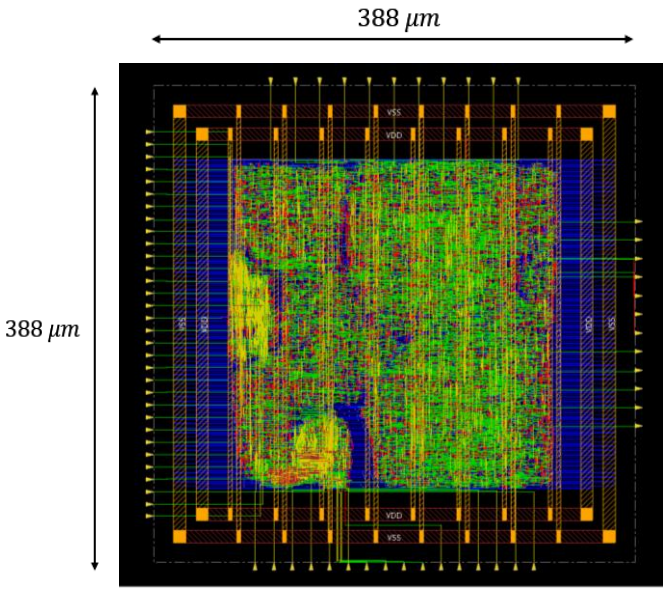

Figure 208 – Tx Placement

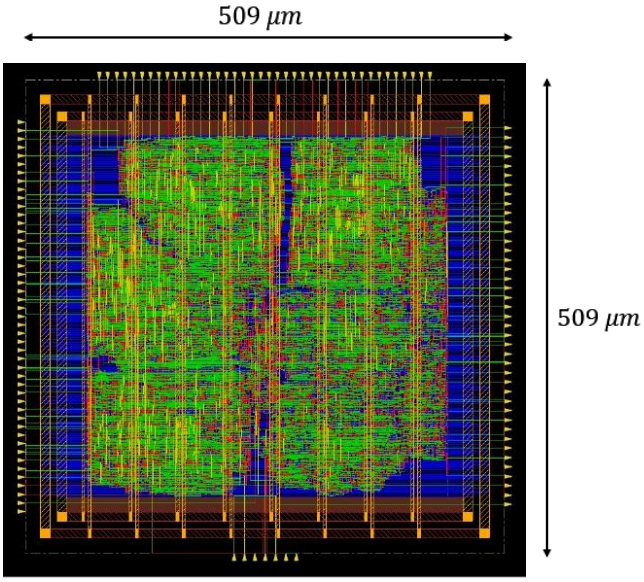

Figure 209 – Rx Placement

4- Pre-CTS Timing Analysis and ECO Optimization

Before Clock Tree Synthesis (CTS), a preliminary timing analysis is conducted based on estimated wire delays from global placement. This stage is essential to identify and fix any violations before clock insertion.

During this phase, the tool performs both setup and hold timing checks. Key timing metrics include:

- **Worst Negative Slack (WNS):** Indicates the most critical timing violation.
- **Total Negative Slack (TNS):** Sum of all timing violations across the design.

For a clean design, both WNS and TNS should be non-negative, and there should be no design rule violations. The analysis results are displayed in the terminal for review.

If negative slack is reported—especially in setup timing—the design must undergo ECO (Engineering Change Order) optimization to correct timing issues. This may include logic restructuring, buffer insertion, or gate sizing to meet timing requirements and prepare the design for CTS.

Setup mode	all	reg2reg	default
WNS (ns):	0.179	9.365	0.179
TNS (ns):	0.000	0.000	0.000
Violating Paths:	0	0	0
All Paths:	4652	2356	2296

Hold mode	all	reg2reg	default
WNS (ns):	-0.095	-0.053	-0.095
TNS (ns):	-53.387	-53.292	-0.095
Violating Paths:	2298	2297	1
All Paths:	4652	2356	2296

Figure 210 – Tx Pre CTS Timing analysis

Setup mode	all	reg2reg	default
WNS (ns):	0.279	0.279	0.346
TNS (ns):	0.000	0.000	0.000
Violating Paths:	0	0	0
All Paths:	15780	9777	10103

Hold mode	all	reg2reg	default
WNS (ns):	-0.077	0.101	-0.077
TNS (ns):	-393.660	0.000	-393.660
Violating Paths:	5797	0	5797
All Paths:	15780	9777	10103

Figure 211 – Rx Pre CTS Timing analysis

5- Clock Tree Synthesis

Clock Tree Synthesis (CTS) is one of the most important stages in PnR. In most of the ICs clock consumes 30-40 % of total power. So efficient clock architecture, clock gating & clock tree implementation helps to reduce power.

It's a technique for distributing the clock equally among all sequential parts of a VLSI design. The purpose of Clock Tree Synthesis is to reduce skew and delay. Clock Tree Synthesis is provided with the placement data as well as the clock tree limitations as input. It balances the clock delay to all clock inputs by inserting buffers/inverters along the clock routes of an ASIC design. As a result, CTS is used to balance the skew and reduce insertion latency. Before CTS, all clock pins were driven by a single clock source. CTS also makes the design meets the clock tree Design Rule Constraints (DRC) like maximum transition delay, maximum load capacitance, and maximum fanout. [46]

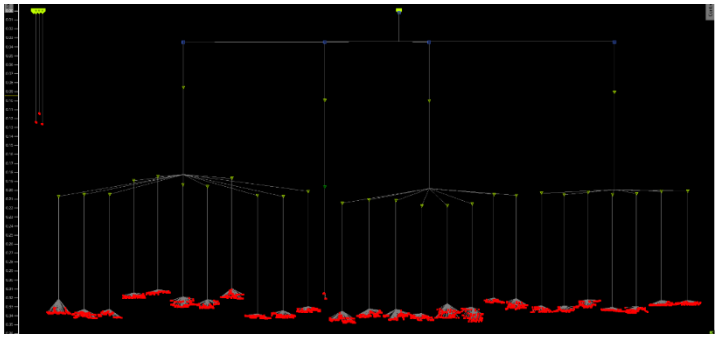

Figure 212 – Tx CTS

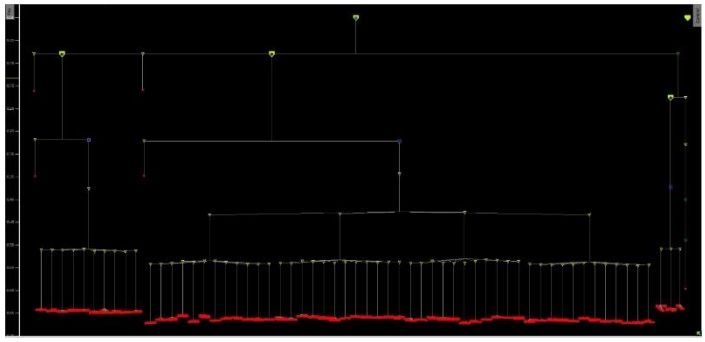

Figure 213 – Rx CTS

Fig. shows CTS (Clock Tree Synthesis) analysis.

- **Red end nodes** represent **registers or sequential elements**, indicating the points where clock edges are captured.
- **Green, yellow, or blue nodes** correspond to **clock buffers or inverters**, which are inserted to **balance clock skew** and **optimize clock delay** across the design.

This visual representation is essential for:

- Identifying **long or imbalanced clock paths** that may cause skew or timing violations.
- Detecting **high fanout segments** that can degrade signal integrity and increase delay.
- Guiding **buffer placement and tree depth adjustments** to enhance timing closure and overall design robustness.

Such analysis ensures that the clock network meets performance and reliability requirements while minimizing power and area overhead.

6- Post CTS timing analysis & ECO optimization

Setup mode	all	reg2reg	default
WNS (ns):	0.191	9.030	0.191
TNS (ns):	0.000	0.000	0.000
Violating Paths:	0	0	0
All Paths:	4652	2356	2296

Hold mode	all	reg2reg	default
WNS (ns):	0.077	0.120	0.077
TNS (ns):	0.000	0.000	0.000
Violating Paths:	0	0	0
All Paths:	4652	2356	2296

Figure 214 – Tx Post CTS timing analysis & ECO

Setup mode	all	reg2reg	default
WNS (ns):	-0.494	0.206	-0.494
TNS (ns):	-3.279	0.000	-3.279
Violating Paths:	10	0	10
All Paths:	15780	9777	10103

Hold mode	all	reg2reg	default
WNS (ns):	0.058	0.087	0.058
TNS (ns):	0.000	0.000	0.000
Violating Paths:	0	0	0
All Paths:	15780	9777	10103

Figure 215 – Rx Post CTS timing analysis & ECO

7- Routing

Routing in the **VLSI design** is making physical connections between signal pins using metal layers. Following Clock Tree Synthesis (CTS) and optimization, the routing step determines the exact pathways for interconnecting standard cells, macros, and I/O pins. The layout creates electrical connections using metals and vias that are determined by the logical connections in the netlist (i.e.; logical connectivity converted as physical connectivity).

What are the steps of routing in VLSI?

Each metal layer in a grid-based routing system has its tracks and preferred routing direction, which are described in a unified cell in the standard cell library. Routing activities are divided into four steps: [47]

1. Global route
2. Track Assignment
3. Detail Routing
4. Search and repair

Post-Routing Verification and Parasitic Extraction

After the **routing stage** is completed, several critical checks and analyses are performed to ensure the integrity and performance of the physical design:

- **Design Rule Check (DRC)** and **Connectivity Check**:
  The routed layout is verified for any **DRC violations** and connectivity issues. If violations are detected, they must be resolved to ensure the layout is manufacturable and functionally correct.
- **RC Extraction**:
  Once the design passes DRC and connectivity checks, **Resistance-Capacitance (RC) extraction** is performed. This process generates a **Standard Parasitic Exchange Format (SPEF)** file, which contains the extracted parasitic resistance and capacitance values from the routed nets.

The SPEF file is then used during **Static Timing Analysis (STA)** and **sign-off simulations** to provide accurate post-layout timing and power analysis. This step is essential as it reflects the **real parasitic effects** introduced by routing, enabling the design to be validated against final performance specifications.

```
#  Command:           verify_drc
##########################################################################

No DRC violations were found
```

Figure 216 – DRC violations

To account for real-world variations in process, voltage, and temperature (PVT) across the chip, On-Chip Variation (OCV) timing analysis is applied. OCV introduces additional timing margins to the analysis by modeling the variation in delay between different paths and elements on the chip. This approach provides a more conservative and realistic timing assessment, helping to identify potential setup and hold violations that may arise under worst-case manufacturing and environmental conditions.

8- Post routing timing analysis & ECO optimization

After performing multiple optimization steps, two hold-violating paths remained unresolved and could not be corrected using standard Engineering Change Order (ECO) optimization techniques.

To address these violations, two buffers were manually inserted along the affected paths to increase the propagation delay and eliminate the hold violations. This was achieved using the following command: ecoAddRepeater

By adding delay through buffer insertion, the arrival time of the signal was aligned with the clock requirements, ensuring that all hold timing constraints were satisfied in the final design.

Setup mode	all	reg2reg	default
WNS (ns):	0.177	9.365	0.177
TNS (ns):	0.000	0.000	0.000
Violating Paths:	0	0	0
All Paths:	4652	2356	2296

Hold mode	all	reg2reg	default
WNS (ns):	0.000	0.000	0.004
TNS (ns):	0.000	0.000	0.000
Violating Paths:	0	0	0
All Paths:	4652	2356	2296

Figure 217 – Tx Post routing timing analysis & ECO optimization

Setup mode	all	reg2reg	default
WNS (ns):	0.062	0.062	0.137
TNS (ns):	0.000	0.000	0.000
Violating Paths:	0	0	0
All Paths:	15780	9777	10103

Hold mode	all	reg2reg	default
WNS (ns):	0.000	0.000	0.002
TNS (ns):	0.000	0.000	0.000
Violating Paths:	0	0	0
All Paths:	15780	9777	10103

Figure 218 – Rx Post routing timing analysis & ECO optimization

9- Chip finishing

The chip finishing step is the final stage before layout sign-off and tape-out. It includes several essential tasks to ensure manufacturability, reliability, and electrical integrity of the chip.

- **Antenna Rule Fixing**:
  Antenna rules define the maximum allowable ratio between the area of metal interconnects and the gate area of transistors to prevent gate oxide damage due to charge accumulation during fabrication. To meet these constraints, two common techniques are used:
    - **Metal jumping**: Rerouting segments on higher metal layers to reduce exposure.

- **Diode insertion**: Adding antenna diodes to provide a safe discharge path for accumulated charge.

- **Filler Cell Insertion**:
  One of the most critical finishing steps is the insertion of filler cells. These are non-functional cells that do not carry any logic. Their purpose is to:
  - Maintain continuity of VDD and GND rails between standard cells, enhancing power distribution.
  - Ensure continuous N-well and substrate connections, which improves substrate biasing and prevents latch-up issues.

- **Metal Fill Insertion**:
  Dummy metal fills are inserted throughout the layout to satisfy metal density requirements imposed by fabrication processes. These fills improve planarity during Chemical Mechanical Polishing (CMP) and enhance manufacturing yield.

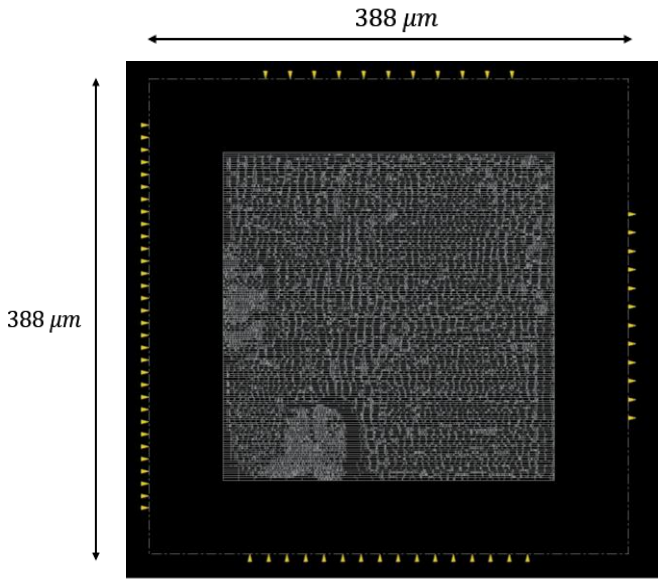

Figure 219 – Tx Chip Layout after Filler Cell

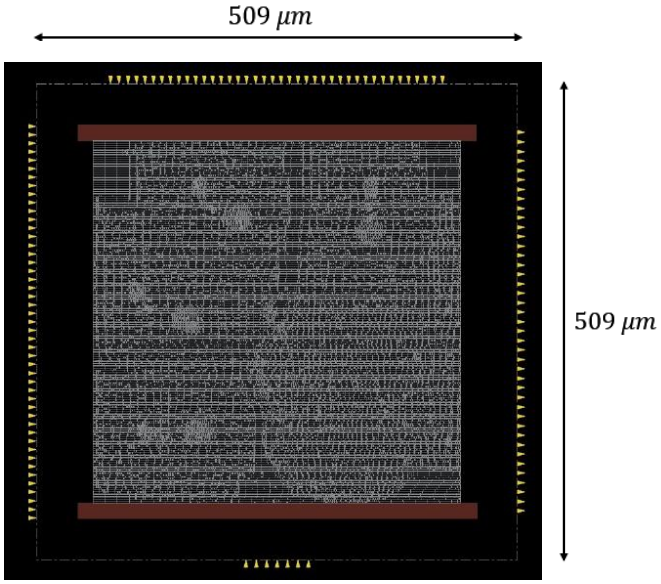

Figure 220 – Rx Chip Layout after Filler Cell

10- GDSII Layout and Files Generation

The final step in the physical implementation flow is the generation of output files.

The physical design tool produces the following output files:

1. **Netlist File (Verilog)**:
   Contains the synthesized gate-level description of the design, representing the final functional implementation.

2. **SDF File (Standard Delay Format)**:
   Used for gate-level simulations to model timing delays accurately based on post-layout information.

3. **SPEF File (Standard Parasitic Exchange Format)**:
   Includes extracted parasitic capacitance and resistance values, enabling precise static timing analysis (STA) and power analysis.

4. **Power and Area Reports**:
   Provide detailed breakdowns of dynamic and static power consumption, along with the total chip area utilized.

5. **GDSII File**:
   Represents the final physical layout of the design and is used for fabrication. It includes all mask layers and geometric data required by the foundry.

These files collectively serve as the handoff package for simulation, sign-off, and silicon manufacturing.

Summarized synthesis results for Tx chain	
Frequency	$1\ GHz$
Total power	$13.26\ mW$
Total area	$55314.72\ \mu m2$

Table 6 – Rx PNR results

Summarized synthesis results for 4 Rx chains	
Frequency	$1\ GHz$
Total power	$5.6449\ mW$
Total area	$110993.76\ \mu m2$

Table 7 – Rx PNR results

Conclusion

In this thesis, a comprehensive design and implementation of a Massive MIMO digital beamforming mm-Wave transceiver for 5G wireless communication was presented. The work began with an extensive literature review that highlighted the evolution of MIMO and beamforming techniques and their critical role in meeting the ever-growing demand for higher data rates, improved spectral efficiency, and enhanced network capacity.

The thesis detailed the principles and types of beamforming, emphasizing digital beamforming as an effective solution for flexible, high-performance beam steering in 5G systems. Various beamforming algorithms, including Delay-and-Sum, MVDR, LCMV, MUSIC, and ESPRIT, were analyzed and compared through MATLAB simulations to validate their performance under different scenarios.

A complete transceiver architecture was then developed, with detailed modeling of both transmitter and receiver chains. Key building blocks such as the Sigma-Delta ADC, Digital Down Converter, Interleaver, Complex Weight Multiplier, Decimation filter, Interpolation filter, and Digital Up Converter were designed, simulated, and integrated into a unified system.

Furthermore, the design was implemented at the Register Transfer Level (RTL) using hardware description languages, followed by logic synthesis, design for testability, formal verification, and place-and-route stages to ensure the design's functionality and manufacturability.

The results demonstrate that the proposed transceiver architecture achieves the desired performance goals, confirming its viability for next-generation 5G networks operating in the mm-Wave band. This work provides a robust foundation for further optimization and potential ASIC fabrication.

References

[1] Marzetta, T. L., Larsson, E. G., Yang, H., & Ngo, H. Q. (2016). Fundamentals of Massive MIMO. Cambridge University Press.

[2] HFCL. (n.d.). Beamforming: Glossary Definition. https://io.hfcl.com/glossary/beamforming

[3] Wikipedia. (n.d.). Phased Array. https://en.wikipedia.org/wiki/Phased_array

[4] PySDR. (n.d.). Direction of Arrival (DOA) estimation. https://pysdr.org/content/doa.html

[5] Berrios, I. T. (2020, August 9). *Introduction to Beamforming (Part 1)*. Medium. https://medium.com/@itberrios6/introduction-to-beamforming

[6] Sanny Telecom. (n.d.). *MIMO Antennas vs. SISO Antennas*. https://www.sannytelecom.com/mimo-antennas-vs-siso-antennas/

[7] Huawei Technologies. (n.d.). *MIMO Overview*. https://info.support.huawei.com/info-finder/encyclopedia/en/MIMO.html

[8] Gao, X., Zhang, Z., & Hu, X. (2018). Massive MIMO: An Overview of Challenges and Opportunities. IEEE Communications Magazine, 56(3), 23–29. https://ieeexplore.ieee.org/abstract/document/8455114

[9] Salehi, N., Ashkezari, M. D., & Kiani, F. (2020). Role of wireless communication technologies in COVID-19 detection and prevention. Journal of Environmental Health Science and Engineering, 18(2), 991–998. https://pmc.ncbi.nlm.nih.gov/articles/PMC7284607/

[10] Wireless Pi. (n.d.). What is the Difference Between Analog, Digital, and Hybrid Beamforming? https://wirelesspi.com/what-is-the-difference-between-analog-digital-and-hybrid-beamforming/

[11] YouTube. (2022, September 25). *Understanding Beamforming for Wireless Systems*. https://youtu.be/9WxWun0E-PM?si=HrSbVpaxWXL1oYjl

[12] Stutzman, W. L., & Thiele, G. A. (2013). *Antenna Theory and Design*. Wiley.

[13] Li, X., & Zhang, Z. (2018). *Comparative Study of High-Resolution Direction-of-Arrival Estimation Algorithms for Array Antenna System*. *Wireless Personal Communications*, 101, 1–12. https://link.springer.com/article/10.1007/s11277-018-6006-9

[14] Cadence Design Systems. (n.d.). Fundamentals of MIMO Communication in Wireless Systems. https://resources.system-analysis.cadence.com/blog/fundamentals-of-mimo-communication-in-wireless-systems

[15] Profolus. (2023). Massive MIMO Explained: Advantages and Disadvantages. https://profolus.com/topics/massive-mimo-explained-advantages-and-disadvantages/

[16] Verkotan. (2023). MIMO, MU-MIMO, and Massive MIMO: Testing at Verkotan. https://verkotan.com/2023/mimo-mu-mimo-and-massive-mimo-mimo-testing-at-verkotan/

[17] Cho, Y. S., et al. (2010). MIMO-OFDM Wireless Communications with MATLAB. Wiley.

[18] Commsbrief. (n.d.). MIMO in LTE: What is Multiple Input Multiple Output in 4G? https://commsbrief.com/mimo-in-lte-what-is-multiple-input-multiple-output-in-4g

[19] ABI Research. (2023). 5G Massive MIMO: Enhancing Data Capacity and User Throughput. https://www.abiresearch.com/blog/massive-mimo

[20] Nokia Bell Labs. (2015). Massive MIMO: An introduction. IEEE Communications Magazine. https://ieeexplore.ieee.org/document/7064850

[21] Verkotan. (2021). Beamforming antennas: How they work and are tested. Verkotan Blog. https://verkotan.com/2021/beamforming-antennas-how-they-work-and-are-tested/

[22] Pao, H., Bai, Y., & Luh, L. (2018). A 16-Element 4-Beam 1 GHz IF 100 MHz Bandwidth Interleaved Bit Stream Digital Beamformer in 40 nm CMOS. IEEE Transactions on Circuits and Systems I: Regular Papers, 65(12), 4215–4227. https://ieeexplore.ieee.org/document/8271864

[23] Hunter Communications. (n.d.). Digital Down Converter (DDC) Theory. https://hunteng.co.uk/support/ddctheory.htm

[24] MathWorks. (n.d.). *Overview of Multirate Filters*. https://www.mathworks.com/help/dsp/ug/overview-of-multirate-filters.html

[25] Jaehun Jeong.(2015).IF-Sampling Digital Beamforming with Bit-Stream Processing (Doctoral dissertation, University of Michigan). Deep Blue. https://deepblue.lib.umich.edu/bitstream/handle/2027.42/116778/jaehun_1.pdf

[26] Khan, M. Z. U., Malik, A. N., Zaman, F., & Qureshi, J. M. (2019). Robust LCMV beamformer for direction of arrival mismatch without beam broadening. Wireless Personal Communications, 104, 21–36. https://doi.org/10.1007/s11277-018-6006-9

[27] CommsBrief. (2021). Difference Between MIMO, SISO, SIMO, and MISO Antenna Systems. https://commsbrief.com/difference-between-mimo-siso-simo-and-miso-antenna-systems/

[28] Wikipedia. (n.d.). *Delta-sigma modulation*. https://en.wikipedia.org/wiki/Delta-sigma_modulation

[29] DTRI. (n.d.). *Sampling rate in LTE and 5G NR baseband systems*. https://www.dtri.in/blogs/post/sampling-rate-in-lte-and-5g-nr-baseband-systems

[30] ShareTechnote. (n.d.). *5G FR Bandwidth*. https://www.sharetechnote.com/html/5G/5G_FR_Bandwidth.html#38_101_2_Table_5_3_5_1

[31] Tretter, S. A. (2008). *Multirate filtering for digital signal processing: MATLAB applications*. Springer.

[32] Hoffmann, J. (1992). *Decimation and interpolation with polyphase filters*. In *IEEE Colloquium on Techniques for Reducing the Complexity of Digital Filters* (pp. 6/1–6/6). https://doi.org/10.1049/ic:19920052

[33] Kumar, S., & Patnaik, A. (2012). *A comparison design of comb decimators for sigma-delta analog-to-digital converters*. International Journal of Computer Applications, 38(1), 20–24.[34] All About Circuits. (n.d.). *Pipelined direct form FIR versus the transposed structure*. https://www.allaboutcircuits.com/technical-articles/pipelined-direct-form-fir-versus-the-transposed-structure/

[35] Li, F., & Zhao, J. (2015). *A novel sigma-delta ADC architecture for low-power biomedical applications*. International Journal of Hybrid Information Technology, 8(6), 213–222. https://gvpress.com/journals/IJHIT/vol8_no6/21.pdf

[36] National Instruments. (n.d.). *Digital upconverter (DUC) - PXIe-5672 features*. https://www.ni.com/docs/en-US/bundle/pxie-5672-features/page/digital-upconverter.html

[37] Hosseini, K., & Kennedy, M. P. (2014). Maximum sequence length MASH digital delta–sigma modulators. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 61(8), 2319–2331.

[38] Bhide, A. (2006). *Design of high-speed time-interleaved delta-sigma D/A converters* (Doctoral dissertation, Delft University of Technology).

[39] Physical Design ASIC. (2020, July). *Process, voltage, temperature (PVT)*. Retrieved from https://physicaldesign-asic.blogspot.com/2020/07/process-voltage-temperature-pvt.html?m=1

[40] Mahapatra, R., & Sinha, R. (n.d.). *ASIC design flow and methodology – An overview*. Retrieved from https://www.researchgate.net/publication/336015572_ASIC_Design_Flow_And_Methodology_-_An_Overview

[41] VLSI Tutorials. (n.d.). *DFT, Scan and ATPG*. Retrieved from https://vlsitutorials.com/dft-scan-and-atpg/

[42] Mahapatra, R., & Sinha, R. (n.d.). *ASIC design flow and methodology – An overview*. Retrieved from https://www.researchgate.net/publication/336015572_ASIC_Design_Flow_And_Methodology_-_An_Overview

[43] Torrubia Ollero, A. (2023). *Content Addressable Memory Implementation for a High-Performance Application* (Master's thesis, Universitat Politècnica de Catalunya). Retrieved from https://upcommons.upc.edu/bitstream/handle/2117/404181/TFM_CAM_Alex_Torrubia_Ollero.pdf?sequence=3&isAllowed=y

[44] Team VLSI. (2021, February). *Pre-placement activities in physical design*. Retrieved from https://teamvlsi.com/2021/02/pre-placement-activities-in-physical-design.html

[45] Team VLSI. (2021, February). *Pre-placement activities in physical design*. Retrieved from https://teamvlsi.com/2021/02/pre-placement-activities-in-physical-design.html

[46] ChipEdge. (n.d.). *What is clock tree synthesis?* Retrieved from https://chipedge.com/what-is-clock-tree-synthesis/#:~:text=Conclusion%3A,power%20in%20most%20integrated%20circuits & https://signoffsemiconductors.com/clock-tree-synthesis-1/

[47] ChipEdge. (n.d.). *What is routing in VLSI physical design?* Retrieved from https://chipedge.com/what-is-routing-in-vlsi-physical-design/