Digital Design and Implementation of Narrowband IoT Physical Downlink Shared Channel Receiver Chain

3GPP REL. 14

“NPDSCH RX R14”

Authors:

Hesham Khaled Abdel-Latif

Mohamed Ammar Mohamed

Reem Saleh Elshawarby

Youssef Ahmed Mohamed Galal

Youssef Taha Esmail

Supervised by:

Dr. Hassan Mostafa

Cairo University

Faculty of Engineering

Department of Electronics and Digital Communications

A thesis submitted for fulfilment of the requirements of Cairo University for Bachelor of Science degree

Acknowledgment

We wish to thank our supervisor: Dr. Hassan Mostafa, for his endless support in resources and encouragement. we are grateful to Si-Vision Inc. and ONE Lab, who supported, planned, and guided this project through all its phases, especially Eng. Khaled Ismail, Eng. Mohamed Maher and Eng. Ramy Raafat for their great efforts to help. Finally, we would like to express gratitude to our professors and TAs for their sincerity in providing their knowledge and expertise to us.

Abstract

Narrowband Internet of Things (NB-IoT) is a new cellular technology introduced in 3GPP Release 13 for providing wide-area coverage for the IoT devices. This thesis provides a hardware implementation of the physical downlink shared channel "NPDSCH" receiver chain in Release 14. We describe how algorithms were constructed and hardware designed in accordance with the requirements of the 3GPP standard to achieve specifications and good performance, low area, low power and low complexity of design. We also provide insight on how the chain was integrated, synthesized, simulated and tested.

Abbreviations

3GPP	3rd Generation Partnership Project
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
CBER	Channel Bit Error Rate
CP	Cyclic Prefix
CDF	Cumulative Density Function
CFO	Carrier Frequency Offset
CORDIC	Coordinate Rotation Digital Computer
ETU	Extended Typical Urban
FFO	Fractional Frequency Offset
ICI	Inter-Carrier Interference
IFO	Integer Frequency Offset
ISI	Inter-Symbol Interference
NPSS	Narrowband Primary Synchronization Signal
PMF	Probability Mass Function
RF	Radio Frequency
SINR	Signal to Interference Noise Ratio
SNR	Signal to Noise Ratio
CRC	Cyclic Redundancy Code or Cyclic Redundancy Check
FFT	Fast Fourier Transform
IFFT	Inverse Fast Fourier Transform
IoT	Internet of Things
LTE	Long-Term Evolution
MSB	Most Significant Bit
NB-IoT	Narrowband Internet of Things
NPBCH	Narrowband Physical Broadcast Channel
NPDCCH	Narrowband Physical Downlink Control Channel

NPDSCH	Narrowband Physical Downlink Shared Channel
NPRACH	Narrowband Physical Random-Access Channel
NPSS	Narrowband Primary Synchronization Signal
NPUSCH	Narrowband Physical Uplink Shared Channel
NRS	Narrowband Reference Signal
NSSS	Narrowband Secondary Synchronization Signal
OFDM	Orthogonal Frequency-Division Multiplexing
QPSK	Quadrature Phase-Shift Keying
RE	Resource Element
TBS	Transport Block Size
UE	User Equipment
WAVA	Wrap Around Viterbi Algorithm
Z-F	Zero-Forcing

Table of Contents

Acknowledgment 2
Abstract 3
Abbreviations 4
1. Introduction 18
1.1. Motivation and Purpose 18
1.2. NB-IoT with Competitors 20
1.3. NPDSCH Receiver based on Rel14. 3GPP 21
1.4. OFDM 22
1.4.1. Cyclic Prefix 22
1.4.2. Advantages of OFDM 22
1.4.3. Disadvantages of OFDM 22
1.5. System Overview 23
1.5.1. Transmission Modes 23
1.5.2. Frame Structure 24
1.5.3. Scheduling in NPDSCH 25
1.6. Project Flow 28
1.7. Design Architecture 28
2. Operation Theory 30
2.1. Coarse Synchronizer 30
2.1.1. Problem Definition 30
2.1.2. Frequency & Time Offsets 30
2.1.3. Operating Environment & Conditions 31
2.1.4. Narrowband Synchronization Signal (NPSS) 31
2.1.5. Acquisition and CFO Extraction Algorithm 32
2.2. CP Remover and Downsampler 34
2.3. CFO Corrector 35
2.3.1. CORDIC Algorithm 35
2.4. FFT Engine 37
2.5. Resource Demapper 38
2.6. Channel Estimation 39
2.6.1. Problem Definition 39
2.6.2. Used channel model 40

2.6.3. Types of Channel Estimation 40
2.6.4. Channel estimation algorithm 41
2.6.5. Least Square Channel Estimation 41
2.6.6. Interpolation 41
2.7. NRS Value Generator 44
2.7.1. Generating the NRS values 44
2.8. NRS Index Generator 45
2.8.1. Generating NRS Indices 45
2.9. Channel Equalizer 46
2.9.1. Complex division 46
2.10. Parallel to Serial and NRS removal 47
2.11. Fine Synchronizer 48
2.11.1. Problem Definition 48
2.11.2. Narrowband Reference Signals NRS 48
2.11.3. Frequency Offset Estimation 48
2.12. Demodulator 50
2.12.1. Problem Definition 50
2.13. Descrambler 51
2.13.1. Problem Definition 51
2.13.2. Polynomials and Equations 51
2.13.3. Descrambler Re-Initialization 52
2.14. Rate De-Matcher 53
2.14.1. Problem Definition 53
2.14.2. The Rate matcher 53
2.14.3. The Rate De-matcher: 55
2.15. Viterbi Decoder 57
2.15.1. Tail Biting Convolutional Encoder 57
2.15.2. Viterbi Algorithm 57
.2.15.3 System Requirements 57
2.15.4. Wrap Around Viterbi Algorithm 58
2.16. Cyclic Redundancy Check 59
3. Digital Design 60
3.1. Coarse Synchronizer 60

3.1.1. Top Module 60
3.1.2. Detailed Hardware 61
3.1.3. Design Challenges and Solutions 61
3.1.4. MATLAB Results 63
3.1.5. Synthesis Results 63
3.2. CP Remover and Downsampler 64
3.2.1. Top Module 64
3.2.2. Synthesis Results 65
3.3. CFO Corrector 65
3.3.1. Top Module 65
3.3.2. Detailed Hardware 67
3.3.3. Design Challenges and Solutions 67
3.3.4. Synthesis Results 67
3.4. FFT Engine 68
3.4.1. Top Module 68
3.4.2. Detailed Hardware 69
3.4.3. Synthesis Results 70
3.5. Resource Demapper 71
3.5.1. Top Module 71
3.5.2. Detailed Hardware 72
3.5.3. Design Challenges and Solutions 73
3.5.4. Results 74
3.6. Channel Estimation 75
3.6.1. Top Module 75
3.6.2. Detailed Hardware 76
3.6.3. Results 77
3.7. NRS Value Generator 79
3.7.1. Top module 79
3.7.2. Detailed Hardware 80
3.7.3. Results 81
3.8. NRS Index Generator 83
3.8.1. Top module 83
3.8.2. Detailed hardware 84

3.8.3. Design Challenges and Solutions 84
3.8.4. Results 85
3.9. Channel Equalizer 86
3.9.1. Top module 86
3.9.2. Detailed Hardware 87
3.9.3. Design Challenges and Solutions 88
3.9.4. Results 88
3.10. Parallel to Serial and NRS removal 89
3.10.1. Top module 89
3.10.2. Design Challenges and Solutions 90
3.10.3. Results 90
3.11. Fine Synchronizer 91
3.11.1. Top Module 91
3.11.2. Detailed Hardware 93
3.11.3. Design Challenges and Solutions 95
3.11.4. Results 95
3.12. Demodulator 96
3.12.1. Top Module 96
3.12.2. Detailed Hardware 97
3.12.3. Results 97
3.13. Descrambler 99
3.13.1. Top Module 99
3.13.2. Detailed Hardware 100
3.13.3. Results 101
3.14. Rate De-Matcher 102
3.14.1. Top Module 102
3.14.2. Detailed Hardware 103
3.14.3. Design Challenges and Solutions 105
3.14.4. Results 105
3.15. Viterbi Decoder 107
3.15.1. Block Interface 107
3.15.2. Detailed Hardware 108
3.15.3. Hardware Challenges and Solutions 110

3.15.4. MATLAB Results 112
3.15.5. Synthesis Results 112
3.16. Cyclic Redundancy Check 113
3.16.1. Top Module 113
3.16.2. Detailed Hardware 114
3.16.3. Results 114
4. Chain Results 115
4.1. Synopsys Design Compiler Synthesis Results 115
4.1.1. Coarse Synchronizer 115
4.1.2. CP Remover and Downsampler 116
4.1.3. CFO Corrector 117
4.1.4. FFT Engine 118
4.1.5. Resource Demapper 119
4.1.6. Channel Estimation 120
4.1.7. NRS Values Generator 121
4.1.8. NRS Index Generator 122
4.1.9. Channel Equalizer 123
4.1.10. Parallel to Serial and NRS removal 124
4.1.11. Fine Synchronizer 125
4.1.12. Demodulator 126
4.1.13. Descrambler 127
4.1.14. Rate De-Matcher 128
4.1.15. Viterbi Decoder 129
4.1.16. Cyclic Redundance Check 130
4.2. DC Results for RX Chain 131
4.2.1. Timing 131
4.2.2. Area Utilization 132
4.2.3. Power Consumption 133
4.3. FPGA Implementation Results 134
4.3.1. Post-Implementation FPGA View 134
4.3.2. Timing 134
4.3.3. Area Utilization 135
4.3.4. Power Consumption: 135

5. FPGA Deployment 136
5.1. FPGA Deployment 136
References 138

List of Figures

Figure [1]: IoT Applications 18
Figure [2]: NB-IoT deployment modes 19
Figure [3]: Data Rates Comparison 20
Figure [4]: Energy Efficiency Comparisons 20
Figure [5]: Guard Time 22
Figure [6]: Guard Time with CP 22
Figure [7]: Frequency/Time Division Duplex Structure 23
Figure [8]: UL/DL in NB-IoT 23
Figure [9]: Frame Structure 24
Figure [10]: Project Flow 28
Figure [11]: NPDSCH Rx Chain Block Diagram 28
Figure [12]: ICI Frequency Offset 30
Figure [13]: NPSS Subframe 31
Figure [14]: CPRDS Finite State Machine 34
Figure [15]: CORDIC Algorithm Rotations 35
Figure [16]: Radix 2^2 Butterfly Diagram 37
Figure [17]: Downlink resource grid 38
Figure [18]: PDP of a multipath system 39
Figure [19]: Channel effect on the signal 41
Figure [20]: Zero-Order interpolation 42
Figure [21]: Interpolation over slot 42
Figure [22]: Interpolation over subframe 42
Figure [23]: BER vs SNR for different types of interpolations 43
Figure [24]: Frame Pilot Locations 48
Figure [25]: QPSK Constellation Diagram 50
Figure [26]: Descrambler Re-initialization 52
Figure [27]: Rate Matcher Subblocks 53
Figure [28]: Interleaver Matrix 54
Figure [29]: Interleavers Intercolumn Permutation Pattern 54
Figure [30]: Interleaver Matrix after performing intercolumn change 54
Figure [31]: Tail Biting Convolutional Encoder for NB-IOT LTE 57
Figure [32]: Trellis Diagram 57
Figure [33]: CRC appending 59
Figure [34]: Coarse Synchronizer Top Module 60
Figure [35]: Coarse Synchronizer Detailed Hardware 61
Figure [36]: LPDT-Operand Isolation 61
Figure [37]: LPDT-Comparison Priority 62
Figure [38]: Window Sliding 63
Figure [39]: NB-IoT One Frame RE Grid 63

Figure [40]: Acquisition and CFO Extraction MATLAB Results 63
Figure [41]: Coarse Synchronizer Synthesis Utilization 64
Figure [42]: CP Remover and Downsampler Top Module 64
Figure [43]: CPRDS Synthesis Utilization 65
Figure [44]: CFO Corrector Top Module 65
Figure [45]: CFO Corrector Detailed Hardware 67
Figure [46]: CFO Corrector Synthesis Utilization 67
Figure [47]: FFT Top Module 68
Figure [48]: FFT Engine Detailed Hardware 69
Figure [49]: Butterfly Hardware Structure 69
Figure [50]: FFT Synthesis Utilization 70
Figure [51]: Resource Demapper Top Module 71
Figure [52]: Resource demapper detailed hardware 72
Figure [53]: Resource Demapper Synthesis result 74
Figure [54]: Channel Estimation Top Module 75
Figure [55]: Channel Estimation detailed hardware 76
Figure [56]: RTL estimates 77
Figure [57]: Matlab estimates 77
Figure [58]: Channel Estimation Synthesis results 78
Figure [59]: NRS Value Generator top module 79
Figure [60]: NRS Value Generator detailed hardware 80
Figure [61]: MATLAB function vs Our function 81
Figure [62]: RTL results 81
Figure [63]: NRS Values Generator Synthesis results 82
Figure [64]: NRS Index Generator top module 83
Figure [65]: NRS Index Generator detailed hardware 84
Figure [66]: MATLAB function vs Our function 85
Figure [67]: RTL results 85
Figure [68]: NRS Index Generator Synthesis results 85
Figure [69]: Channel Equalizer top module 86
Figure [70]: Channel Equalizer detailed hardware 87
Figure [71]: RTL equalized OFDM symbol 88
Figure [72]: Matlab equalized OFDM symbol 88
Figure [73]: Equalizer Synthesis results 88
Figure [74]: Parallel to Serial and NRS removal top module 89
Figure [75]: Parallel to Serial and NRS removal Synthesis results 90
Figure [76]: Fine Synchronization Top Module 91
Figure [77]: Fine Synchronization Detailed Design 93
Figure [78]: Arctan Linear Range 94
Figure [79]: Fine Synchronizer RTL Results 95
Figure [80]: Fine Synchronizer Synthesis Results 95

Figure [81]: Demodulator Top Module 96
Figure [82]: Demodulator Detailed Hardware 97
Figure [83]: Demodulator MATLAB Results 97
Figure [84]: Demodulator RTL Results 97
Figure [85]: Demodulator Synthesis Results 98
Figure [86]: Descrambler Top Module 99
Figure [87]: Descrambler Detailed Design 100
Figure [88]: Descrambler MATLAB Results 101
Figure [89]: Descrambler RTL Results 101
Figure [90]: Descrambler Synthesis Results 101
Figure [91]: Rate De-Matcher Control Finite State Machines 104
Figure [92]: Rate De-Matcher MATLAB Results 105
Figure [93]: Rate De-Matcher RTL Results 105
Figure [94]: Rate De-Matcher Synthesis Results 106
Figure [95]: Decoder Interface 107
Figure [96]: Detailed hardware of the decoder 108
Figure [97]: Trellis Diagram 108
Figure [98]: Example of Path Metric Unit operation 109
Figure [99]: Finite State Machine implemented in the control unit 109
Figure [100]: Hardware added to avoid overflow 110
Figure [101]: Another implementation of traceback operation to reduce memory size 111
Figure [102]: MATLAB Implementation of WAVA Results 112
Figure [103]: Viterbi Decoder Synthesis Utilization 112
Figure [104]: CRC Top Module 113
Figure [105]: CRC Detailed Design 114
Figure [106]: CRC MATLAB Results 114
Figure [107]: CRC RTL Results 114
Figure [108]: CRC Synthesis Results 114
Figure [109]: Coarse Synchronizer DC Area Results 115
Figure [110]: Coarse Synchronizer DC Power Results 115
Figure [111]: CP Remover and Downsampler DC Area Results 116
Figure [112]: CP Remover and Downsampler DC Power Results 116
Figure [113]: CFO Corrector DC Area Results 117
Figure [114]: CFO Corrector DC Power Results 117
Figure [115]: FFT Engine DC Area Results 118
Figure [116]: FFT Engine DC Power Results 118
Figure [117]: Resource Demapper Area report 119
Figure [118]: Resource Demapper Power report 119
Figure [119]: Channel Estimation Area report 120
Figure [120]: Channel Estimation Power report 120
Figure [121]: NRS Values Generator Area report 121

Figure [122]: NRS Values Generator Power report....121
Figure [123]: NRS Index Generator Area report....122
Figure [124]: NRS Index Generator Power report....122
Figure [125]: Channel Equalizer Area report....123
Figure [126]: Channel Equalizer Power report....123
Figure [127]: Parallel to Serial and NRS removal Area report....124
Figure [128]: Parallel to Serial and NRS removal Power report....124
Figure [129]: Fine Synchronization Area Report....125
Figure [130]: Fine Synchronization Power Report....125
Figure [131]: Demodulator Area Report....126
Figure [132]: Demodulator Power Report....126
Figure [133]: Descrambler Area Report....127
Figure [134]: Descrambler Power Report....127
Figure [135]: Rate De-Matcher Area Report....128
Figure [136]: Rate De-Matcher Power Report....128
Figure [137]: Viterbi Decoder Area Reports....129
Figure [138]: Viterbi Decoder Power Reports....129
Figure [139]: CRC Area Report....130
Figure [140]: CRC Power Report....130
Figure [141]: Timing Report....131
Figure [142]: Area Utilization Histogram....132
Figure [143]: Area Utilization DC Report....132
Figure [144]: Power Consumption Histogram....133
Figure [145]: Total Power DC Report....133
Figure [146]: Post-Implementation FPGA View....134
Figure [147]: Vivado Timing Report....134
Figure [148]: FPGA Area Utilization....135
Figure [149]: FPGA Power Consumption Summary....135
Figure [150]: FPGA Deployment Schematic....136
Figure [151]: RX Chain Output....137
Figure [152]: BER vs SNR Curve....137

List of Tables

Table [1]: ETU Channel Tabs and Gains 32
Table [2]: Cyclic Prefix Lengths 34
Table [3]: CPRDS State Table 34
Table [4]: CORDIC Micro-Rotation Steps 36
Table [5]: FFT Radii Complexity 37
Table [6]: Power Delay Profile of ETU channel 40
Table [7]: Channel Estimation Algorithms 41
Table [8]: In phase and Quadrature components of QPSK 50
Table [9]: Coarse Synchronizer Interface Table 60
Table [10]: CP Remover and Downsampler Interface Table 64
Table [11]: CFO Corrector Interface Table 65
Table [12]: FFT Engine Interface Table 68
Table [13]: Resource Demapper Interface Table 71
Table [14]: Ordering table 74
Table [15]: Channel Estimation Interface Table 75
Table [16]: NRS Value Generator interface table 79
Table [17]: NRS Index Generator interface table 83
Table [18]: Channel Equalizer interface table 86
Table [19]: Parallel to Serial and NRS removal interface table 89
Table [20]: Fine Synchronization Interface Table 91
Table [21]: Demodulator Interface Table 96
Table [22]: Descrambler Interface Table 99
Table [23]: Rate De-Matcher Interface Table 102
Table [24]: Decoder Interface Table 107
Table [25]: CRC Interface Table 113
Table [26]: Area Utilization Table 132
Table [27]: Total Power Consumption Table 133

This page was intentionally left blank

Introduction CHAPTER 01

1.1. Motivation and Purpose

The number of wireless devices is rapidly increasing. So, we want to connect these devices together to able to develop useful applications. This connectivity can be done through IoT.

The main concept of IoT is to have a network of wireless devices to collect and exchange data between them over the internet or any communication network. IoT doesn't require the device, software or sensor to be connected to public internet it should only be connected to a specific network and can be addressed.

There is a lot of IoT applications such smart cities, traffic management, smart metering and many more.

These IoT devices and network should follow certain specifications, as the devices should be of low cost and long battery life and the network should support low latency and coverage requirements for wide area.

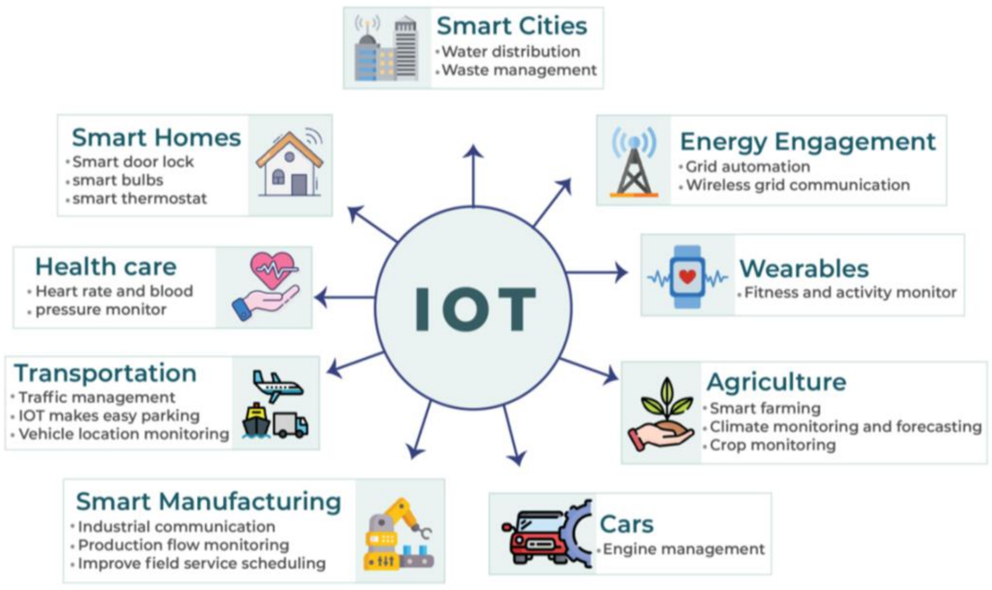


Figure [1]: IoT Applications

NB-IoT is a new mobile network that only used for IoT applications and it's based on LTE. NB-IoT supports a range of data rate depending on the channel quality and the bandwidth besides it offers energy saving capabilities to increase battery life and it support.

It is designed to achieve the perfect co-existence performance along with other technologies. For NB-LTE Rel.14 it works with one resource block occupying a bandwidth of 180 KHz.

NB-IoT is designed to support three deployment modes

1. Standalone mode

   An NB-IoT carrier is deployed independently of any LTE carrier and it can function as replacement for GSM carrier. Besides it also provides deployment flexibility based on spectrum availability.
2. Guard band

   An NB-IoT carrier is deployed within the guard band of an LTE carrier by using a used Resource Block within LTE carrier Guard Band. This doesn't consume any capacity from the primary LTE traffic carrier
3. In band

   An NB-IoT carrier is deployed occupying a physical resource block (PRB) within an LTE carrier. This mode is the most efficient one as it allows the base station schedule to multiplex LTE and NB-IoT traffic in the same spectrum.

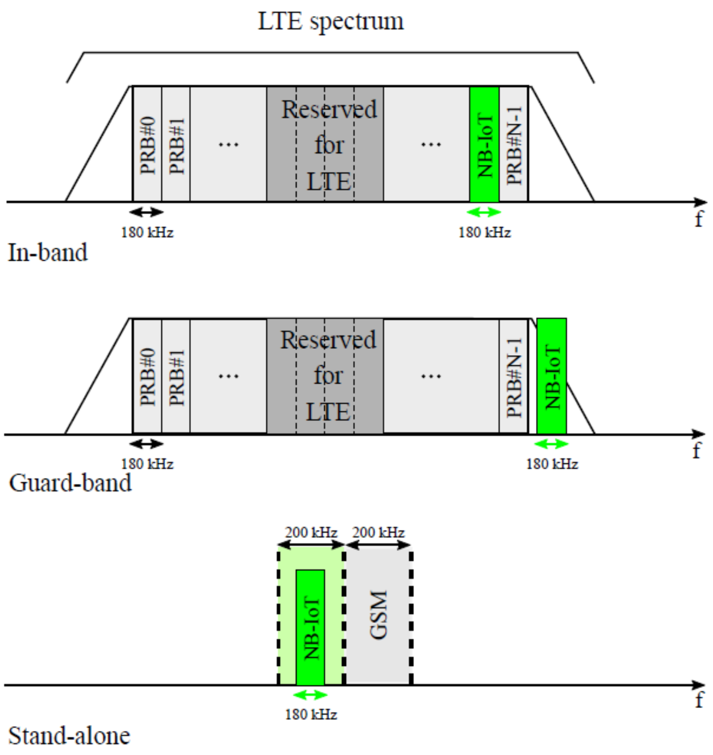


Figure [2]: NB-IoT deployment modes

1.2. NB-IoT with Competitors

Low Power Wide Area Network LPWAN connects wireless devices designed for Internet of Things IoT and manages to combine efficient energy devices with wide transmission ranges so they can run for a very long time on battery-based systems.

There are three famous technology nodes using LPWA, NB-IoT, Long Range Wide Area Network (LoRaWAN) and Sigfox. So, what makes NB-IoT special? NB-IoT is based on Long Term Evolution LTE, so that makes it an open standard not proprietary as LoRaWAN and Sigfox so they are used in.

NB-IoT is also a leading technology in terms of data throughput, data rates and energy efficiency. As the chart shows. LoRaWAN suffers from interference in ISM band which reduces data rates while Sigfox connection is lost in poor conditions.

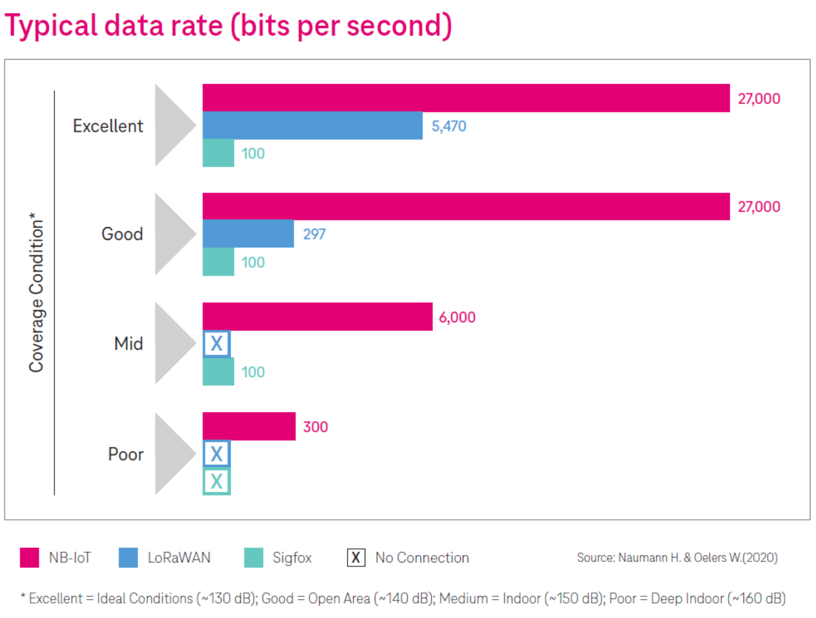


Figure [3]: Data Rates Comparison

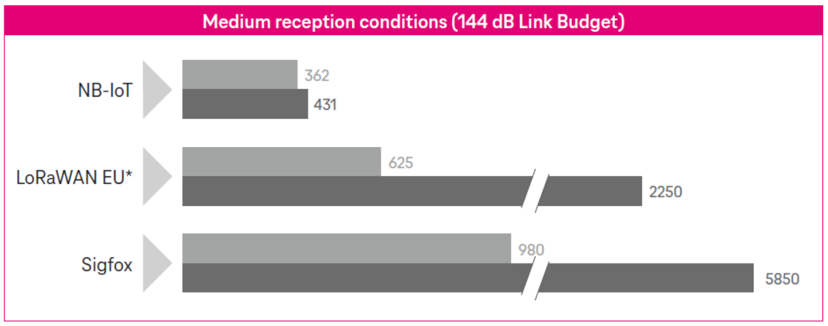


Figure [4]: Energy Efficiency Comparisons

1.3. NPDSCH Receiver based on Rel14. 3GPP

3GPP first introduced the NB-LTE in release 13 and further enhancements were made later in release 14 to improve the Quality-of-Service QoS. And also increased the transport block size up to 2536 bits instead of being 680 in release 13.

1.4. OFDM

Orthogonal Frequency Division Multiplexing (OFDM) is a method of data transmission in which information is split among several closely spaced narrowband subchannel frequencies instead of using wide band frequency channel.

The spacing is precisely chosen so as to provide orthogonality which gives us immunity to interference and letting the demodulators blind to any other frequency than the intended one.

1.4.1. Cyclic Prefix

As we need to be more immune to fading effects so, we need to cancel the Inter Symbol Interference (ISI). This is done by adding a guard time. This time is chosen larger than the expected delay spread in order to a multipath component of one symbol can interfere with the next symbol.

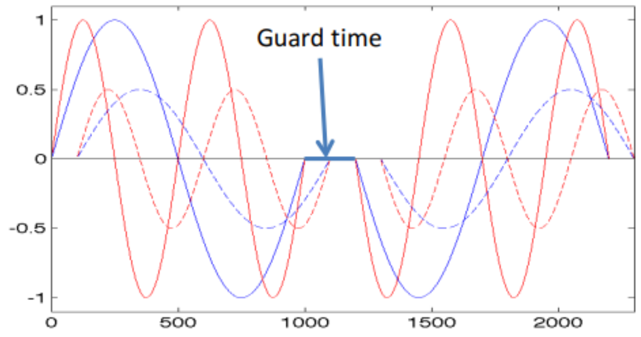


Figure [5]: Guard Time

Making this guard time consists of no signal give us a problem which is Inter Carrier Interference (ICI). ICI is the crosstalk between different subcarrier meaning that we lost orthogonality.

We eliminate this ICI by making OFDM symbols cyclically extended in the guard time by this we maintain the orthogonality by ensuring that delayed replicas of the OFDM symbol always have integer number of cycles within the FFT interval.

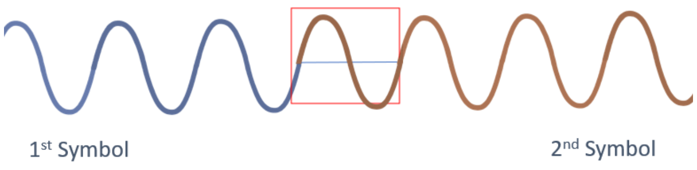


Figure [6]: Guard Time with CP

1.4.2. Advantages of OFDM

- OFDM efficiently deals with multipath fading as for a given delay spread the complexity is significantly lower than that of single carrier system.
- Data rates and number of subcarriers can adaptively change according to channel quality.
- Unlike FDM there is no guard bands here so it gives better spectral utilization

1.4.3. Disadvantages of OFDM

With all the previous advantages OFDM offers some challenges to be solved as it very sensitive to frequency offset and doppler shifts. Hence, we need to correct these offsets to able to use OFDM reliably.

1.5. System Overview

1.5.1. Transmission Modes

LTE Supports 3 types of frame structures

- Type 1 for Frequency Division Duplex (FDD)
- Type 2 for Time Division Duplex (TDD)
- Type 3 for LAA secondary cell operation

The main difference between FDD and TDD is that FDD uses different frequency bands for the uplink and downlink transmission while TDD uses the same frequency band but with time slots dedicated for uplink and other for downlink transmission.

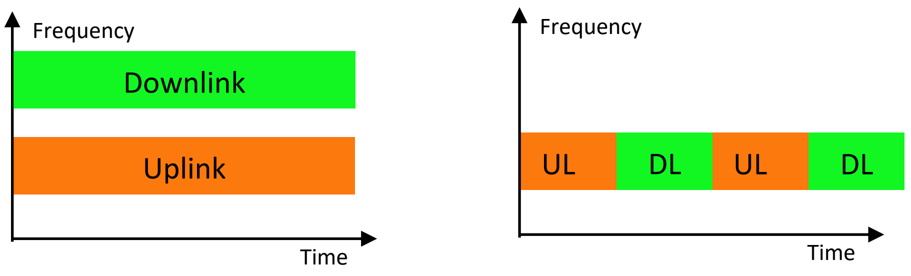


Figure [7]: Frequency/Time Division Duplex Structure

In NB-IoT, frame structure type 1 is used, here is a closer look on how NB-Iot uplink and downlink transmition takes place.

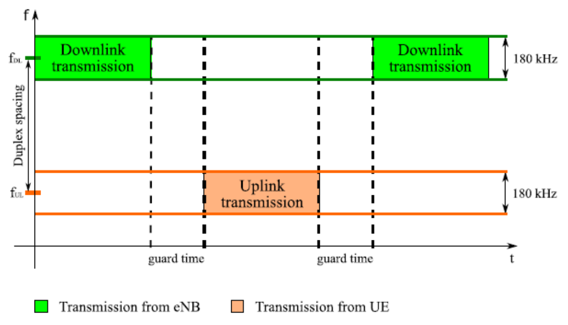


Figure [8]: UL/DL in NB-IoT

1.5.2. Frame Structure

Each NB-IoT radio frame has duration of 10 ms and consists of 10 subframes with 1 ms duration each. Every subframe consists of 2 slots of 0.5 ms each. The slot consists of 7 OFDM symbols. And the frequency band of 180 kHz is divided into 12 sub carriers of 15 kHz each.

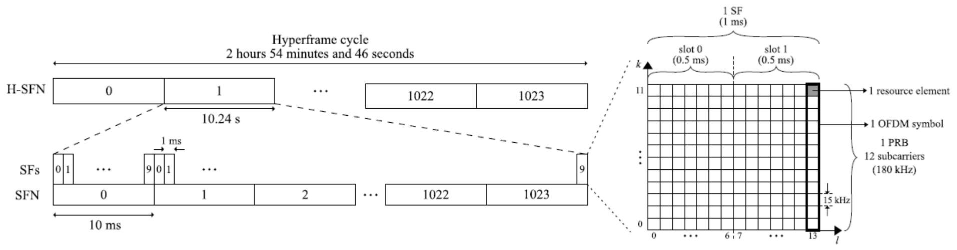
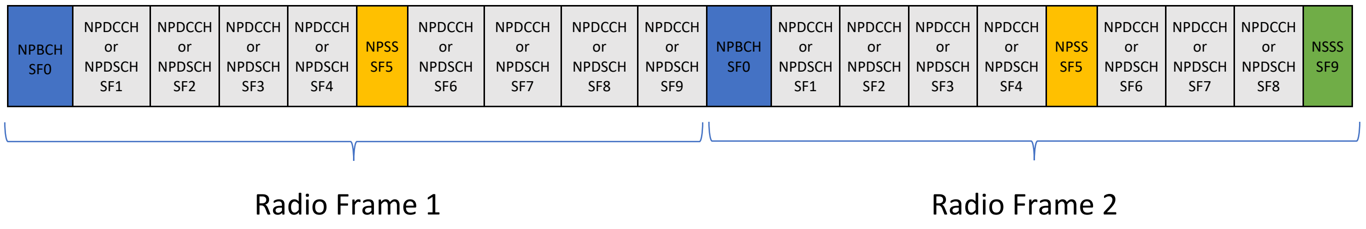


Figure [9]: Frame Structure

There are subframes that carries different types of information such as the broadcast subframes and the Control Subframes the synchronization subframes such as the Narrowband Primary Synchronization Signal NPSS and the Narrowband Secondary Synchronization Signal NSSS.



NPBCH SF0	NPDCCH or NPDSCH SF1	NPDCCH or NPDSCH SF2	NPDCCH or NPDSCH SF3	NPDCCH or NPDSCH SF4	NPSS SF5	NPDCCH or NPDSCH SF6	NPDCCH or NPDSCH SF7	NPDCCH or NPDSCH SF8	NPDCCH or NPDSCH SF9	NPBCH SF0	NPDCCH or NPDSCH SF1	NPDCCH or NPDSCH SF2	NPDCCH or NPDSCH SF3	NPDCCH or NPDSCH SF4	NPSS SF5	NPDCCH or NPDSCH SF6	NPDCCH or NPDSCH SF7	NPDCCH or NPDSCH SF8	NSSS SF9

Radio Frame 1
Radio Frame 2

According to 3GPP

Frame time $T_f = 307200T_s = 10ms, T_{subframe} = 30720T_s = 1ms$

$$T_s = \frac{1}{15000 \times 2048} ms$$

$T_{cp} = 5.2\ \mu s$ for 1st OFDM symbol and $4.7\mu s$ for the rest

$T_u = \frac{1}{15K} = 66.67\mu s$ and $T_{symbol} = \frac{1ms}{14} = 71.4\mu s$

Number of samples for cp = 160 for the first ODFM symbol and 144 for the rest. So, the minimum number of samples used in LTE is 128 to get integer number of samples for $cp = 10$ for the first OFDM symbol and 9 for the rest.

For NB-IoT LTE it follows the number of minimum samples which is 128 although its number of subcarriers is 12. Hence, we use down sampler in our design from 128 to 16.

So, the sampling rate $T_s = \frac{1}{(15000)(128)} = \frac{1ms}{1.92} = 520ns$

1.5.3. Scheduling in NPDSCH

Scheduling is handled by the downlink control information DCI. In case of not carrying system information block SIBs-NB the NPDSCH carries user data towards the UE and it takes place after transmission of NPDCCH in order for the receiver to be able to decode the DCI.

The DCI of the NPDSCH is DCI format type N1 for CRC not masked with RA-RNTI which is random access response and it consists of:

Field	Number of bits	Description
Flag for format N0/format N1 differentiation	1	0 for N0, 1 for N1
Subcarrier order indicator	1	
Scheduling delay I_{delay}	3	indicates the start of the codeword
Resource Assignment I_{SF}	3	indicates the number of subframes
Modulation and coding scheme I_{MCS}	4	indicates the transport block size
Repetition number I_{Rep}	4	indicates the number of repetitions
New data indicator	1	
HARQ-Ack resource	4	
DCI subframe repetition number	2	
Total number of bits	23	

1. Scheduling delay I_{delay}

   The scheduling delay is indicating the start of the codeword successive subframes after $n + 5 + k_o$ subframes, the NPDCCH that ends in n subframes so usually the NPDSCH follows the NPDCCH after 5 or more subframes.

I_{delay}	k_o	
	$R_{max} < 128$	$R_{max} \geq 128$
0	0	0
1	4	16
2	8	32
3	12	64
4	16	128
5	32	256
6	64	512
7	128	1024

2. Resource Assignment I_{SF}

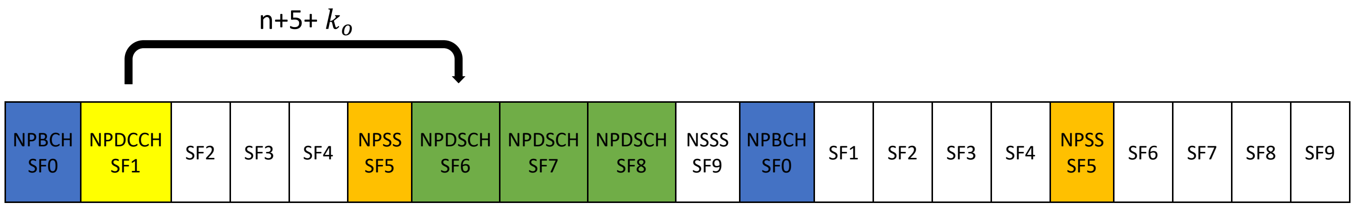
   The resource assignment indicates the number of subframes that is constituting the NPDSCH message.

I_{SF}	N_{SF}
0	1
1	2
2	3
3	4
4	5
5	6
6	8
7	10

Example:

$I_{delay} = 0,\ k_o = 0$

$I_{SF} = 2\ ,N_{SF} = 3$



3. Modulation and coding scheme I_{MCS}

   The modulation and coding scheme in case of NPDSCH not carrying system information block SIBs-NB is the same as I_{TBS} where $I_{TBS} = I_{MCS}$ indicates the transport block size that is used in transmission and it is an important parameter in order to decode the data correctly and hand it over from the physical layer to upper layer. There's no segmentation in NB-IoT because the maximum TBS is 2536 which is less than 6144.

I_{TBS}	I_{SF}							
	0	1	2	3	4	5	6	7
0	16	32	56	88	120	152	208	256
1	24	56	88	144	176	208	256	344
2	32	72	144	176	208	256	328	424
3	40	104	176	208	256	328	440	568
4	56	120	208	256	328	408	552	680
5	72	144	224	328	424	504	680	872
6	88	176	256	392	504	600	808	1032
7	104	224	328	472	584	680	968	1224
8	120	256	392	536	680	808	1096	1352
9	136	296	456	616	776	936	1256	1544

10	144	328	504	680	872	1032	1384	1736
11	176	376	584	776	1000	1192	1608	2024
12	208	440	680	904	1128	1352	1800	2280
13	224	488	744	1032	1256	1544	2024	2536

4. Repetition number I_{Rep}
   The repetition number indicates the number of repetitions of the codeword.

I_{Rep}	N_{Rep}
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	192
9	256
10	384
11	512
12	768
13	1024
14	1536
15	2048

1.6. Project Flow

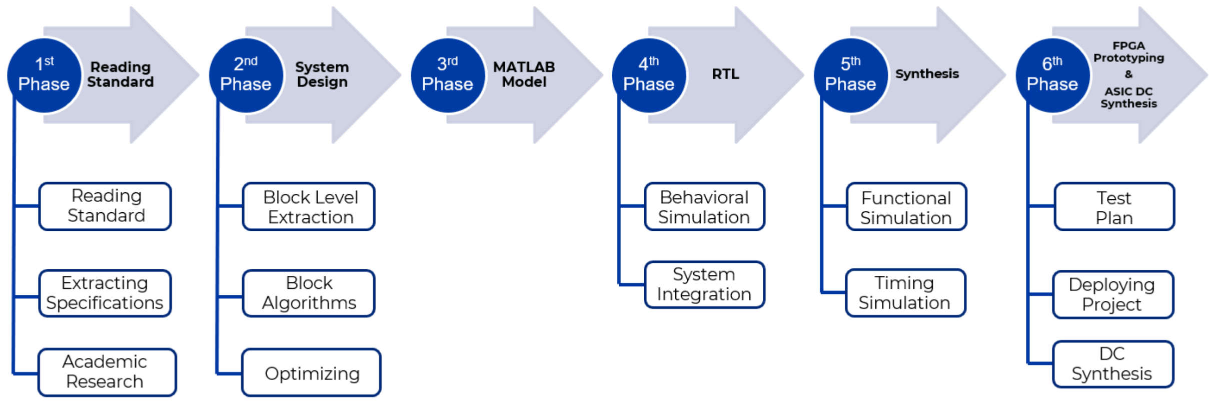


Figure [10]: Project Flow

1.7. Design Architecture

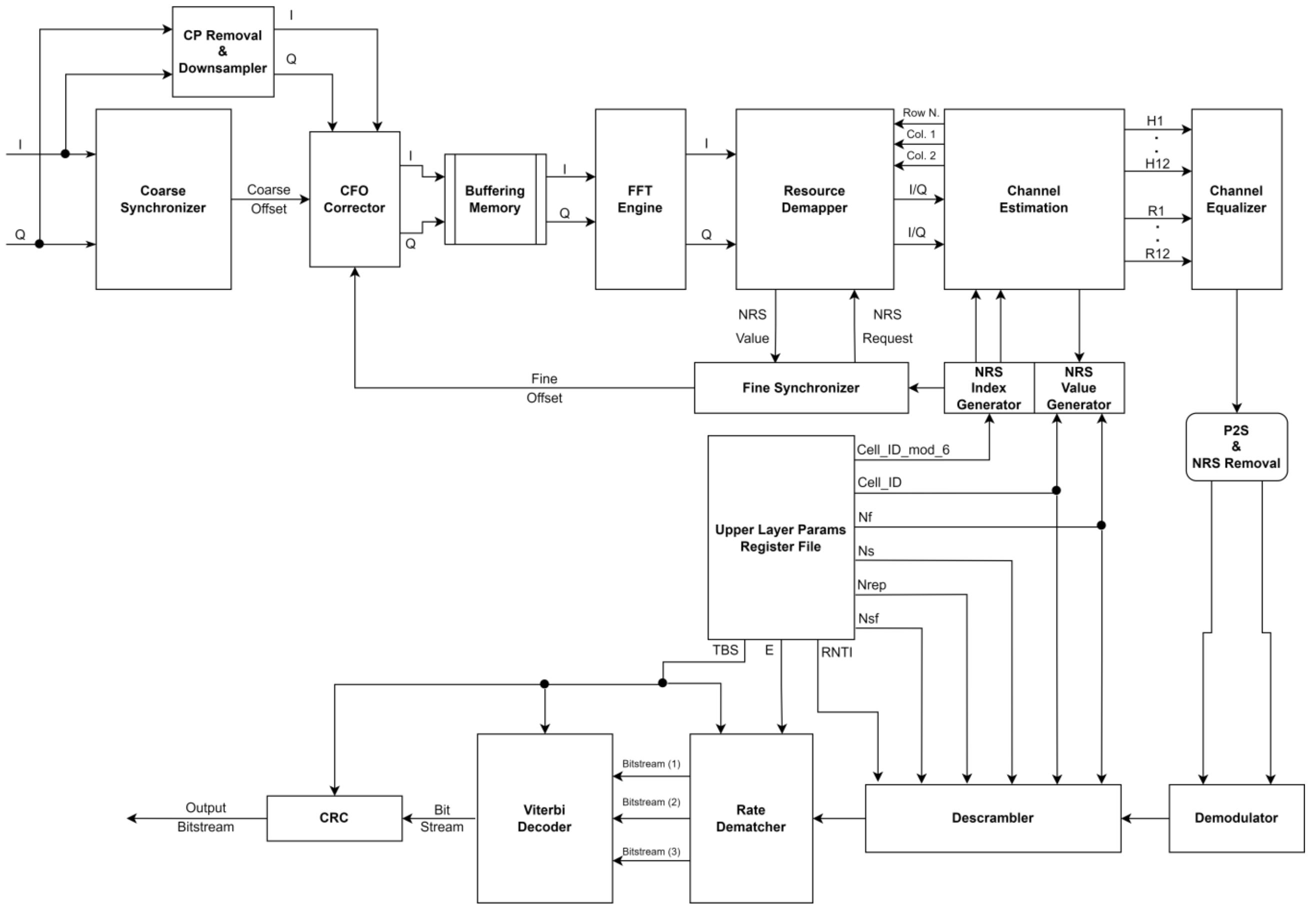


Figure [11]: NPDSCH Rx Chain Block Diagram

The in-phase and quadrature phase data enter the receiver chain through the Coarse Synchronizer.

After the synchronization happen, we remove the cyclic prefix using CP Removal and we down sample the data.

Then these data pass through CFO Corrector to correct the frequency offset using the estimated frequency offset given by the Coarse Synchronizer.

Fine Synchronization is used to keep tracking of time and frequency offset.

There is a FIFO based memory used to store samples to prepare them for the FFT which will transform these samples from time domain to frequency domain.

The Resource Demapper then stores the QPSK symbols output from the FFT till it have a full subframe. It used to ensure storing the subframe constant during processing time.

The Channel Estimation then extract the pilots from the subframe using NRS Index Generator and with the use of NRS Value Generator it tries to estimate the channel effect and send these estimates to the Channel Equalizer to compensate this effect.

Data then pass through the Parallel to Serial and NRS Remove block to remove the pilots and convert the parallel to serial data stream.

This QPSK symbol stream consists of real and imaginary data pass through the demodulator to convert it into bits based on the constellation then goes to the descrambler to restore the scrambled data at transmitter.

The bit stream finally goes through the Rate De-matcher to improve the channel efficiency by changing the code rate of the transmitted data.

Then data pass through the Viterbi decoder which is used to decode the data encoded in convolutional code at the transmitter side then finally reach CRC block to get an acknowledgement signal.

2.1. Coarse Synchronizer

2.1.1. Problem Definition

OFDM based systems faces two challenging problems, ICI (Inter-Carrier Interference) and ISI (Inter-Symbol Interference). ICI problem arises from the basic assumption of the subcarrier orthogonality, this assumption means that the spectrum of an arbitrary signal on an arbitrary subcarrier must have nulls at all other subcarriers. Thus, a frequency offset introduced from the channel will inevitably introduce ICI as shown in figure (12).

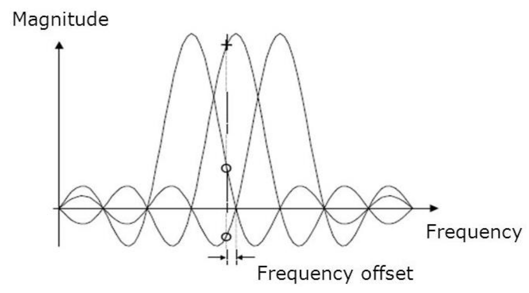


Figure [12]: ICI Frequency Offset

ISI problem isn't exclusive for OFDM based systems, as it takes place when two symbols interfere with each other in time domain because of multipath propagation dispersion, poor pulse shaping at the transmitter or a synchronization error at the receiver. Both problems cause a degradation in the overall system performance and SINR. Thus, these problems will be addressed later in order to solve them.

2.1.2. Frequency & Time Offsets

Regarding ICI, it has two main sources;

1st source is Carrier Frequency Offset (CFO) which is the difference between the carrier frequency at the transmitter and the RF oscillators at the receiver. CFO can take a value up to 18KHz assuming a maximum mismatch of 20ppm between the transmitter and receiver oscillators and a carrier frequency around 900 MHz.

2nd source is the raster offset. When a UE turns on it conducts a search in frequency domain looking for a carrier to facilitate the synchronization process, referred to as "anchor carrier", this carrier is searched for on a 100 KHz raster, so the offset between the 100 KHz raster and the center frequency of the anchor carrier is defined as the raster offset. the NB-IoT's cell search and acquisition are designed for the UE to be able to synchronize having up to 7.5KHz raster offset. Assuming that sampling oscillator and carrier oscillator are unified, thus, Sampling Frequency Offset (SFO) will be accounted for if we considered the CFO.

Thus, the total maximum frequency offset is assumed to be ± 25.5 KHz.

Considering ISI, for this context, perfect pulse shaping is assumed; also, it is assumed that the cyclic prefix (CP) is longer than the multipath channel's maximum excess tap delay. Thus, errors in estimating the beginning of symbols in the synchronization process are the source of ISI of interest in this context.

2.1.3. Operating Environment & Conditions

Due to coverage enhancement in the 14th release of 3GPP on NB-IoT, the minimum SNR when assuming a maximum coupling loss of 164dB is -12.6dB in a guard-band or an in-band deployment. The multipath fading channel is modeled by the Extended Typical Urban (ETU) model, which is considered the worst-case channel, having 9 taps with a maximum excess tap delay of 5 us and a maximum Doppler frequency of 5 Hz due to the stationary nature of the applications of NB-IoT.

2.1.4. Narrowband Synchronization Signal (NPSS)

The NPSS is a subframe that is sent along every NB-IoT frame and used by the devices to achieve synchronization, in both time and frequency, to an NB-IoT cell. The NPSS needs to be designed so that it is detectable even with a very large frequency offset. Because of the consideration of device complexity required for NPSS detection, all the cells in an NB-IoT network uses the same NPSS. figure (13) below shows the NPSS signal within NB-IoT frame.

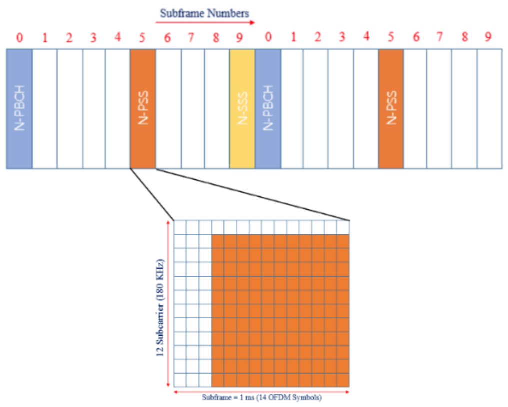


Figure [13]: NPSS Subframe

NPSS base waveform in time domain is generated from a Zadoff-Chu (ZC) sequence of a root index equals to 5 in frequency domain occupying 11 subcarriers of an OFDM symbol in 11 OFDM symbols. It's required from a ZC sequence to work on a prime number. Thus, the 12th subcarrier and first three OFDM symbols are left unused. The sequence is generated by:

$$d(l) = S(l).e^{\frac{-j\pi un(n+1)}{11}} \rightarrow \ u = 5 \ \ , \ n = 0,1,2,\ldots,10$$

$$S(l) = \{1\,,1,1,1,-1,-1,1,1,1,-1,1\}$$

Where $S(l)$ is the 3GPP standardized code cover.

The reason behind the choice of ZC sequence is that it guarantees zero-crossing auto/cross correlation when correlated with any different signal and a maximum value when correlated with itself so it satisfies the Constant Amplitude Zero Autocorrelation (CAZAC) property which limits the Peak to Average Power Ratio (PAPR) and provides ideal cyclic autocorrelation.

2.1.5. Acquisition and CFO Extraction Algorithm

2.1.5.1. Channel Modelling

The channel is modelled to be Rayleigh ETU Fading channel with AWGN noise. According to standard release 14, ETU can be modelled with channel tabs and gains of

Table [1]: ETU Channel Tabs and Gains

Channel Tabs (ns)	0	50	120	200	230	500	1600	2300	5000
Channel Gains (dB)	-1	-1	-1	0	0	0	-3	-5	-7

The received time domain signal at the receiver should be modelled as:

$$r(n) = \left(\left(x[n] \circledast \mathrm{h[n]} \right) * e^{-j2\pi.\epsilon.n.Ts} \right) + w[n]$$

where: $x[n]$ is the baseband signal, $h[n]$ is the channel impulse response, exponent models the CFO (ϵ) where $\epsilon = \epsilon_f + \epsilon_I$, $w[n]$ represents the AWGN noise.

2.1.5.2. Initial Acquisition Stage

Slide and correlate algorithm is used to calculate the auto-correlation metric for a single frame period with a window of 10 symbols. This is made by multiplying each OFDM symbol to the conjugate of the proceeding symbol (sample by sample). This is done after multiplying each sample with the NPSS code cover to correct symbols that might have been multiplied by “-1” in the transmitter.

$$R(k) = \sum_{i=1}^{Nwindow} \left(r(i).S\left(\frac{i}{Ns}\%11\right)\right).\left(\left(r(i+Ns).S\left(\frac{i}{Ns}+1\right)\%11\right)\right)^*$$

where:

- k: Represents the sample shift
- N_{window}: Number of samples per window
- N_s: Number of samples per symbol
- i: Sample iterator for the same sample shift (k)
- $R(k)$: Auto-Correlation Metric for different sample shifts (k)

For hardware memory utilization purposes, we can't afford having a memory to store this huge metric, thus, it is reduced by a factor of 16, which is done by adding each consecutive 16 windows, resulting in a memory of just 1200 locations instead of 19200 locations.

$$R_{reduced}(k) = \sum_{i=16k}^{16(k+1)-1} R(k)$$

Due to low SNR environments mentioned earlier, this metric has to be averaged over many consecutive frames (M) so it can a reliable outcome. Nevertheless, due to this low SNR, stochastic random peaks might appear in the metric, so, to suppress those peaks we will apply a running average filter on the reduced metric value using the following TD Filter. Where (m) covers 1200 locations only.

$$A(m) = A(m).(1-\alpha) + R_r(m).(\alpha) \quad \rightarrow \quad 0 < \alpha < 1$$

Coarse Timing and FFO can be estimated then using:

$$\tau = argmax\ \forall_m(|A| * 16) - 8$$

$$\epsilon_f = \frac{N}{2\pi N_s}\, angle\left(A\left(\frac{\tau+8}{16}\right)\right)$$

This decision is taken only when the absolute value of the metric of any index exceeds a specific threshold that we extracted by simulating over 500 LTE Frames.

2.1.5.3. CFO Extraction and Fine-Tuning Stage

This stage is done by cross-correlating a clean locally generated version of NPSS with the incoming time domain received signal.

$$C(\tau_c, \epsilon_c) = \sum_{i=\tau_c}^{\tau_c+N_p-1} r(i).\left(p(i-\tau_c).e^{\frac{-j2\pi\epsilon_c(i-\tau_c)}{N}}\right)^*$$

Where;

- $r(i)$ is received time domain signal
- $p(i)$ is locally generated NPSS
- $\tau_c = [\tau - \delta, \tau + \delta]$
- $\epsilon_c = ([-2 \sim 2] * 14\ KHz) + \epsilon_f$

Deltas in Coarse Timing is a design parameter, chosen to start from 16 until it reaches 1 with the following offsets (±8, ±4, ±2). Number of averaged frames in 2nd stage is also a design parameter chosen to be 5 frames for each internally tuning stage. Thus, Final Coarse Timing and CFO can be calculated using

$$(\tau_c, \epsilon_c) = argmax\forall_{\tau_c,\epsilon_c}|C|$$

2.2. CP Remover and Downsampler

Baseband transmitters upsamples the baseband signal from 16 point to 128 point to increases resolution, improves anti-aliasing filter performance and reduces noise. Nevertheless, after upsampling time domain signal baseband transmitters uses cyclic prefix in Frequency Division Multiplexing schemes including OFDM to primarily act as a guard band between successive symbols to overcome inter-symbol interference, ISI. Use of cyclic prefix is a key element of enabling the OFDM signal to operate reliably. This is done by copying the last 9 and/or 10 samples to the OFDM symbol start in all NB-IoT subframes according to table (2) below:

Table [2]: Cyclic Prefix Lengths

Symbol #	1	2	3	4	5	6	7	8	9	10	11	12	13	14
CP Length	10	9	9	9	9	9	9	10	9	9	9	9	9	9

So, we need to remove cyclic prefix and then downsample the time domain signal by a factor of 8. We approached this by using a Finite State Machine (FSM) of the following states;

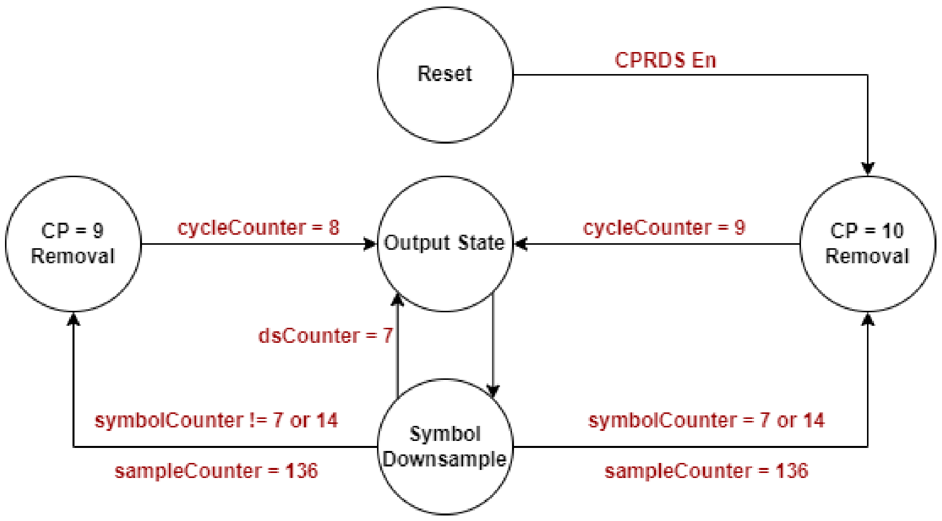


Figure [14]: CPRDS Finite State Machine

Table [3]: CPRDS State Table

State	**Description**
Reset	Considered as idle state until start of operation
CP10 Removal	Used for 1st and 8th symbols CP Removal
CP9 Removal	Used for rest of subframe symbols CP Removal
Downsample	Used to skip samples in each symbol
Output	Outputs one TD sample in every 8 samples

2.3. CFO Corrector

2.3.1. CORDIC Algorithm

CFO Corrector is based on CORDIC Algorithm which is used to correct time introduced frequency offset by successively rotating time domain symbols with fixed predetermined angles. CORDIC algorithm is an iterative approach for generating trigonometric functions such as sine and cosine that uses rotations to calculate a wide range of elementary functions using simply shift and add.

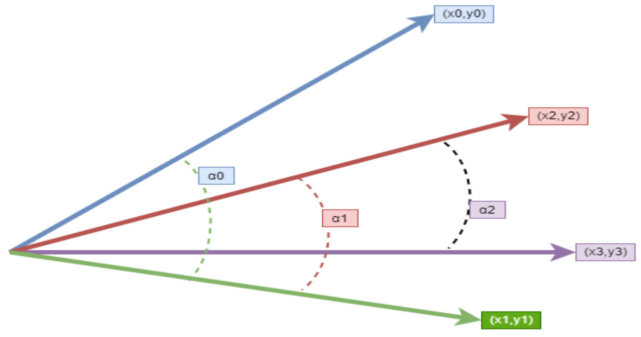


Figure [15]: CORDIC Algorithm Rotations

So, to rotate a vector with a given angle θ using CORDIC algorithm as shown in Figure(X) we rotate the vector with constant angles α_i using the following rotation matrix:

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \begin{pmatrix} \cos\alpha_i & -\sin\alpha_i \\ \sin\alpha_i & \cos\alpha_i \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix}$$

$$x_{i+1} = x_i \cos\alpha_i - y_i \sin\alpha_i$$

$$y_{i+1} = x_i \sin\alpha_i + y_i \cos\alpha_i$$

$$\theta_{i+1} = \theta_i - d_i\alpha_i$$

Where d_i is the sign of θ_i. So, after n iteration the $\theta_n = 0$, we can take $\cos\alpha_i$ common factor we get:

$$x_{i+1} = \cos\alpha_i (x_i - y_i \tan\alpha i)$$

$$y_{i+1} = \cos\alpha_i \,(y_i + x_i \tan\alpha_i)$$

$$\theta_{i+1} = \theta_i - d_i\alpha_i$$

To simplify the implementation of the CORDIC algorithm we chose $\tan \alpha_i$ to be equals to 2^{-i} to implement the multiplication using shifting operations so the final form of equations, where d_i is the sign of θ_i:

$$x_{i+1} = \cos \alpha_i (x_i - d_i y_i 2^{-i})$$

$$y_{i+1} = \cos \alpha_i \left(y_i + d_i x_i 2^{-i}\right)$$

$$\theta_{i+1} = \theta_i - d_i \alpha_i$$

So, after n iterations the new vector will be as follows:

$$\begin{pmatrix} x_n \\ y_n \end{pmatrix} = \cos \alpha_1 * \cos \alpha_2 * \dots * \cos \alpha_n \begin{pmatrix} 1 & -\tan \alpha_1 \\ \tan \alpha_1 & 1 \end{pmatrix} \begin{pmatrix} 1 & -2^{-1} \\ 2^{-1} & 1 \end{pmatrix} \begin{pmatrix} 1 & -2^{-2} \\ 2^{-2} & 1 \end{pmatrix} \dots \begin{pmatrix} 1 & -2^{-n} \\ 2^{-n} & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$$

So, we can pre calculate the value of the multiplication of the $\cos \alpha_i$ as it will be constant k

$$x_n = k(x_0 - d_n y_0 2^{-n})$$

$$y_n = k(y_0 + d_n x_0 2^{-n})$$

$$\theta_n = 0$$

The set of angles that will be used are in range $-99.7 \leq \theta \leq 99.7$ as the sum of all angles obeying the law of $\tan \alpha_i$ equals to 2^{-i} is 99.7

Table [4]: CORDIC Micro-Rotation Steps

$\tan \alpha$	α	$\cos \alpha$
1	45	0.707106781
0.5	26.5650511771	0.894427191
0.25	14.0362434679	0.9701425
0.125	7.1250163489	0.992277877
0.0625	3.5763343750	0.998052578
0.03125	1.7899106082	0.999512078
0.015625	0.8951737102	0.999877952
0.0078125	0.4476141709	0.999969484
0.00390625	0.2238105004	0.999992371
0.001953125	0.1119056771	0.999998093
0.000976562	0.0559528919	0.999999523
0.000488281	0.0279764526	0.999999881
0.00024414	0.0139882271	0.99999997

From the table we can compute the value of k = 0.607252941

2.4. FFT Engine

FFT Engine is used for time domain processing, where it takes 16-point time domain samples and transforms them into frequency components (QPSK Symbols) for farther processing later on.

We chose the Radix 2^2 algorithm as it has the same multiplicative complexity as radix 4 algorithm (which is lower than radix 2), but retains the butterfly structure of radix 2 algorithm, which is very suitable for simple and efficient hardware implementation. As shown below in table (5)

Table [5]: FFT Radii Complexity

Radix	# Complex Additions	# Complex Multiplications	Hardware Complexity
Radix-2	64	17	Simple
Radix-4	96	9	Complex
Radix-2^2	64	8	Simple

Single Delay Line Feedback (SDF) architecture is used instead of memory based because SDF architecture has very much lower latency as it saves the memory accessing time in both read/write cycles. SDF architectures also minimize the required memory, which can dominate circuit area and power dissipation. Figure (X) below shows Radix 2^2 butterfly diagram;

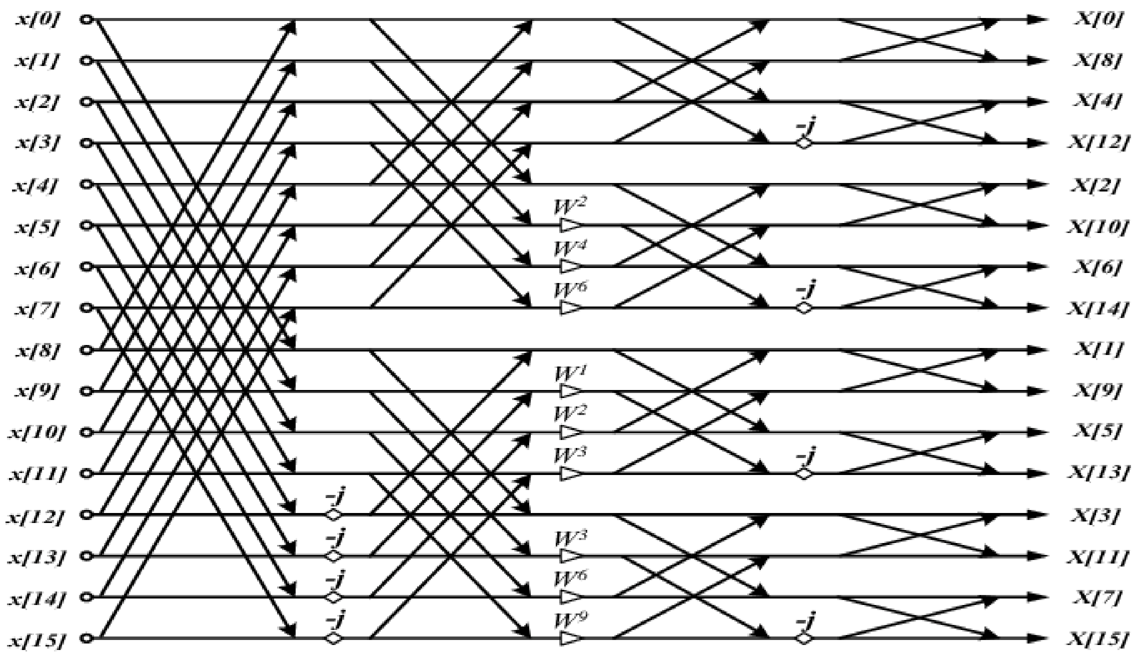


Figure [16]: Radix 2^2 Butterfly Diagram

2.5. Resource Demapper

Each frame is 10ms in time and consisting of 10 subframe, 1ms each. These subframes are divided into two slots of 0.5ms each. This slot is known as our resource block with 7 OFDM symbols.

A resource block represents one time slot consisting of seven consecutive OFDM symbols $N_{symb}^{DL} = 7$ and twelve subcarriers $N_{sc}^{RB} = 12$, giving us a total of $N_{symb}^{DL} \times N_{sc}^{RB}$ of resource elements.

A resource block is (RB) is the smallest unit of resources that can be allocated to a user. It is used to represent mapping of certain physical channel to a resource element.

As in NB-LTE we use one RB with normal cyclic prefix (CP) giving $\Delta f = 15KHz$. Hence, each resource element is $15KHz$ wide giving us $BW = 12 * 15 = 180KHz$

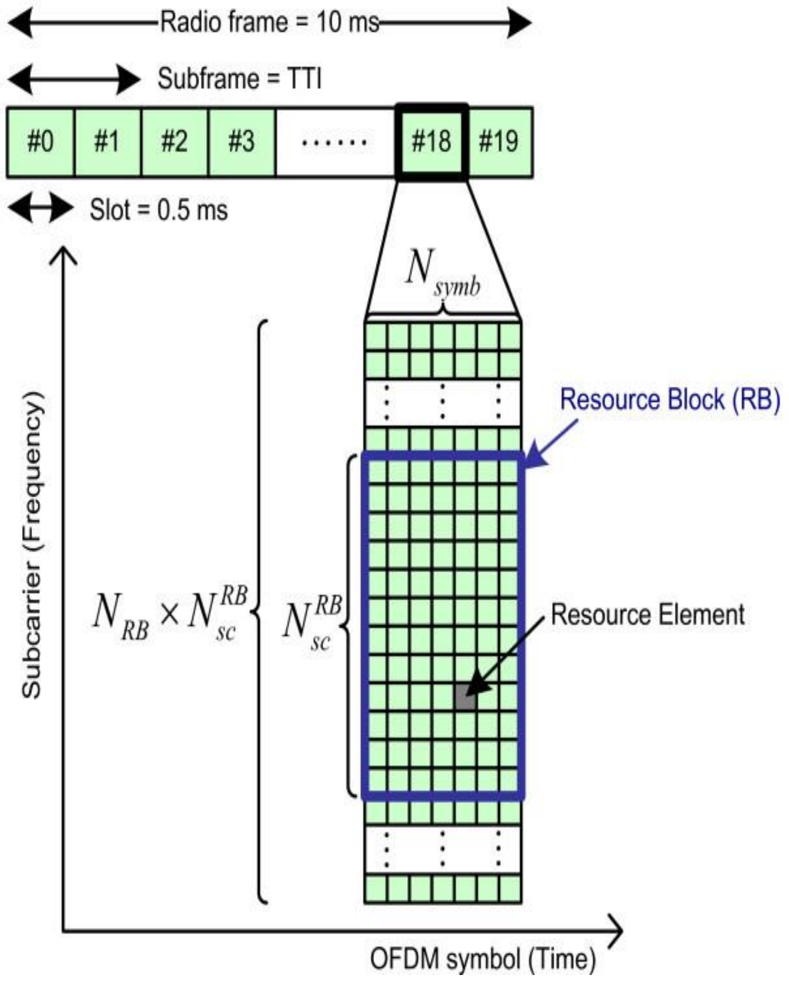


Figure [17]: Downlink resource grid

2.6. Channel Estimation

2.6.1. Problem Definition

As we deal with wireless channel, we need to take into consideration the environment of our channel and model it properly.

Any wireless channel would suffer from fading which can be classified into

1- Large scale fading
2- Small scale fading

Large Scale Fading:

It is for the large variation found in the received signal amplitude it mainly happens because of

Path Loss:

It is a reduction in the power of the wave as it propagates through the space and it's inversely proportional to the distance.

Shadowing:

It is caused by the obstacles in the path of propagation between Tx and Rx

Small Scale Fading:

Caused by interference between versions of the transmitted signal which arrive at the receiver in slightly different times. This interference can be constructive or destructive and as it happening between slightly delayed version so, it may result in rapid variation in phase or amplitude.

Mobile communication also suffers from which is called High-Speed train conditions which arise from moving of the user equipment (UE) while receiving the signal.

Power Delay Profile:

It gives the intensity of the signal received through a multi-path fading channel as function of the time delay. This time delay is the difference in travel time between the multipath arrivals.

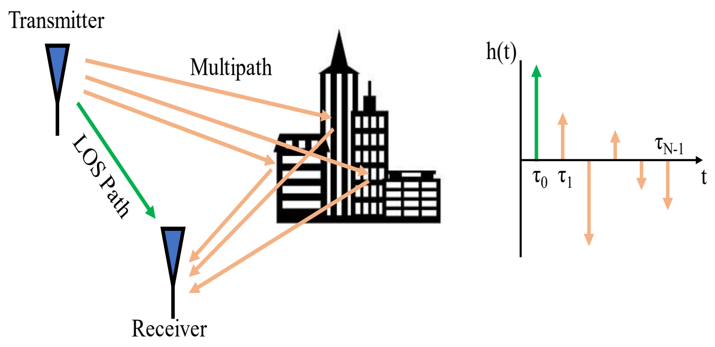


Figure [18]: PDP of a multipath system

2.6.2. Used channel model

We used a multipath fading channel model as the channel is modeled as Extended Typical Urban (ETU) with the following delay profile as specified by 3GPP standard.

Table [6]: Power Delay Profile of ETU channel

Excess tap delay (ns)	Relative power (dB)
0	-1
50	-1
120	-1
200	0
230	0
500	0
1600	-3
2300	-5
5000	-7

Maximum doppler frequency shift is 5 Hz and with sampling frequency 1.92 MHz

2.6.3. Types of Channel Estimation

There are mainly three types to estimate the channel effect on the signal to be able to compensate it.

There are mainly three types of channel estimation

Pilot based: transmitter send certain symbols, which is known to the receiver, to be used in channel estimation process

Blind: it uses statistics from the incoming data to estimate the channel effect without the need of known symbols

Semi-blind: it uses both statistics from sent data as well as pilots to estimate the channel effect

The channel estimation of the Narrowband Physical Downlink Shared Channel (NPDSCH) is a pilot-based process in which transmitter sends some known symbol and the receiver generate a noiseless version of these symbol then tries to find the relation between the locally generated and the received pilots.

The pilots are called as Narrowband Reference Signal (NRS) there are certain equations and sequence generators to calculate its value and location as per the standard.

These pilots are found in every subframe except for the Narrowband Primary Synchronization Signal (NPSS) and Narrowband Secondary Synchronization Signal (NSSS).

The NRS are found in four subcarriers in each slot with constant magnitude of $\frac{1}{\sqrt{2}}$.

2.6.4. Channel estimation algorithm

Pilot based channel estimation used in NB-IoT has many algorithms with a tradeoff between performance with complexity and requirements of low power and area.

Table [7]: Channel Estimation Algorithms

Algorithm	**Complexity**
Least square	$O(N)$
Minimum Mean Square (classic)	$O(N^3)$
Minimum Mean Square (improved)	$O(N^2)$

We used least square method for its low complexity then used interpolation to compensate for accuracy.

2.6.5. Least Square Channel Estimation

Assume received signal y that is equal

$$y = xH + AWGN$$

x: transmitted signal

H: channel effect

$AWGN$: additive white gaussian noise

Lease square channel estimated $Hest$

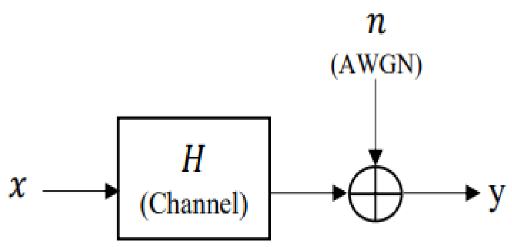


Figure [19]: Channel effect on the signal

$$Hest = \frac{y}{x}$$

By that we see that the algorithm neglects the effect of the additive white gaussian noise which indicates less accuracy but it is less complexity also.

The block aims to effectively divide the received pilot over the locally generated pilots to get the channel effect and then interpolation is done to have better estimate for the channel after that the estimated channel values are given to the equalizer.

2.6.6. Interpolation

As the pilots are found in only four subcarriers while we have twelve subcarriers so interpolation process must be done.
Interpolation can be done in three different techniques

2.6.6.1. Zero order interpolation

Here the interpolation is constant in which each three subcarriers having one pilots will be divided by its channel estimate. H_{LS}

The channel estimation gives four estimates per slot as there are four NRS. Each estimate will be used by three consecutive subcarriers.

This is the simplest one but with lowest performance.

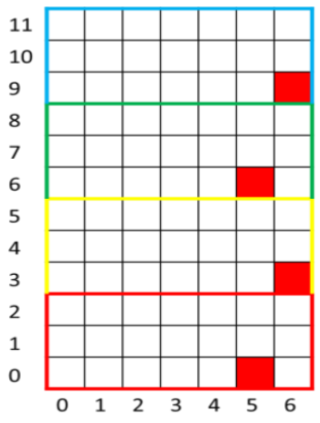


Figure [20]: Zero-Order interpolation

2.6.6.2. Interpolation over slot

Linear interpolation is calculated over the slot, in each slot we have four estimates we use them and interpolate giving twelve different outputs H_{LS}each one used by the equalizer with its corresponded subcarrier.

More complex than the previous with slight increase in the performance.

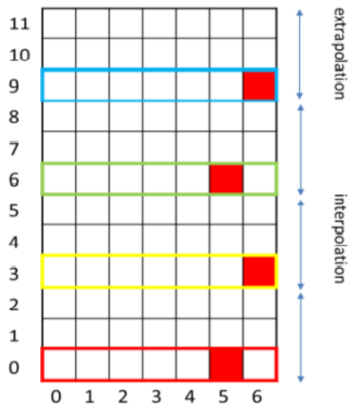


Figure [21]: Interpolation over slot

2.6.6.3. Interpolation over subframe

Linear interpolation is calculated over the subframe, in each subframe we have two slots giving a total of eight estimates.

Averaging first is done over every two on the same subcarrier then liner interpolation over the twelve subcarriers giving twelve different estimates to be used by the equalizer.

The most complex of the three but with noticeable impact on the performance.

This is the one to be used in our system.

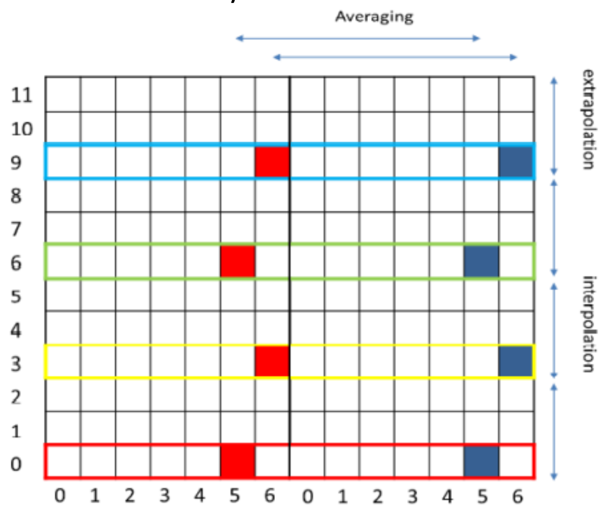


Figure [22]: Interpolation over subframe

2.6.6.4. Comparison

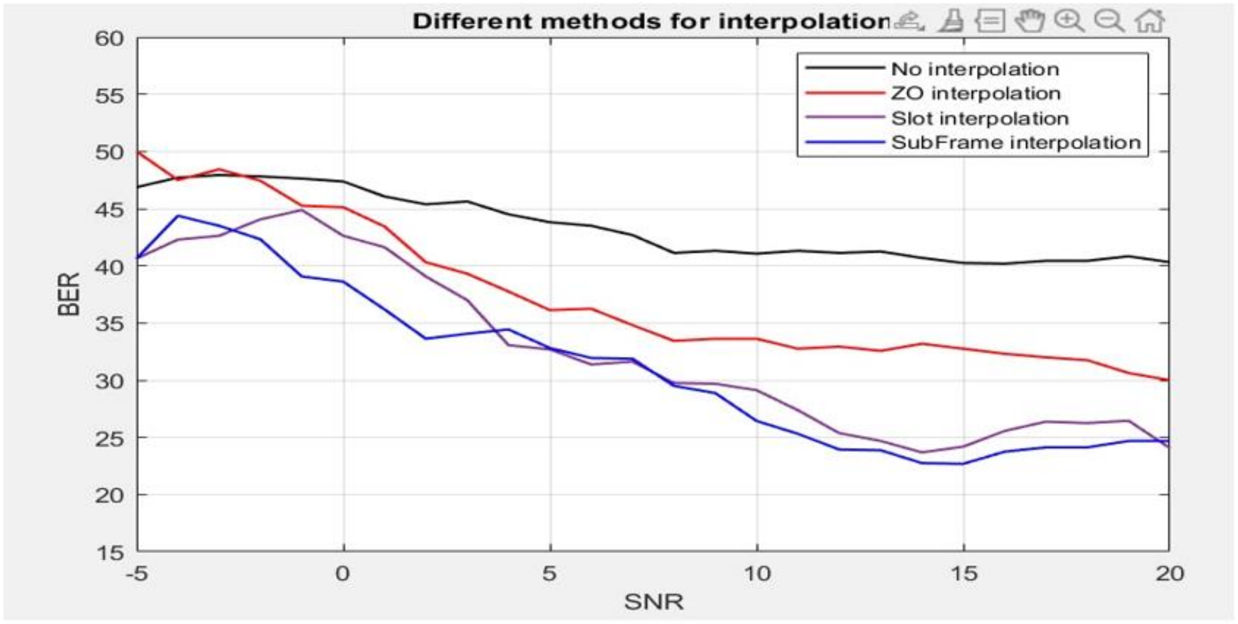


Figure [23]: BER vs SNR for different types of interpolations

2.7. NRS Value Generator

NRS are our pilots which will be used to estimate the channel effect so, we need to locally generate these pilots. According to 3GPP standard there are specific equations to get the values of our pilots.

2.7.1. Generating the NRS values

2.7.1.1. Variables

NRS values change every slot depending on three variables

l: $number\ of\ OFDM\ symbol\ carrying\ the\ NRS$ (5,6)
m: m = 0,1, … ,$2NRB\ maxDL$ − 1 → $so\ m$ = 0,1

Where N_{RB}^{maxDL} is the maximum downlink resource block which is equal to 1 in narrowband case.

ns: $number\ of\ slot\ ranges\ from$ 0 to 19

2.7.1.2. Equations

$$r_{l,ns}(m) = \frac{1}{\sqrt{2}}\,(1 - 2C(2m)) + \frac{j}{(\sqrt{2})}(1 - 2C(2m + 1))$$

Where C refers to pseudo random sequence generated by 31-length golden sequence defined below

$$C(n) = (x1\,(n + Nc\,) + x2\,(n + NC\,))mod2$$

And sequences $x1$and $x2$ defined as follow:

$$x1\,(n + 31) = (x1\,(n + 3) + x1\,(n))mod2$$

Initialized with $x1\,(0) = 1$ and $x1\,(n) = 0\ for\ n = 1,2,3,..,30$

$$x2\,(n + 31) = (x2\,(n + 3) + x2\,(n + 2) + x2\,(n + 1) + x2\,(n))mod2$$

Initialized with sequence c_{init}

$$cinit = 2\,10(7(ns + 1) + l + 1)(2N_{ID}^{cell} + 1) + 2N_{ID}^{cell} + Ncp$$

And

Nc = 1600

N_{ID}^{cell} ∈ (0,504) and it’s input from upper layer.

Ncp = 1 for normal cyclic prefix

2.8. NRS Index Generator

There are set of defined equation in the standard for the NRS location as their location vary, we must track it to extract it from the resource block.

2.8.1. Generating NRS Indices

2.8.1.1. Variables

NRS location within the resource block depend on

$v: variable\ which\ change\ with\ OFDM\ symbol\ number$

$v_{shift}: which\ depend\ on\ N_{ID}^{cell}$

$m: m = 0,1, \ldots, 2N_{RB}^{maxDL} - 1 \rightarrow so\ m = 0,1$

2.8.1.2. Equations

$$k = 6m + (v + v_{shift})$$

$N_{symb}^{DL}: number\ of\ symbols\ in\ one\ slot\ which\ is$ 7

$l = N_{symb}^{DL} - 2, N_{symb}^{DL} - 1 = 5,6$ in each slot

$k: refers\ to\ the\ subcarrier\ carrying\ the\ pilot$

So, it can be said that k refers to the number of rows in which the pilot is placed within the resource block

$v = 0,\ if\ l = 5$

$v = 1,\ if\ l = 6$

$v_{shift} = N_{ID}^{cell}\ mod 6$

Note: $N_{ID}^{cell}\ mod6$ is assumed to be given from upper layer

2.9. Channel Equalizer

The data transmitted through wireless medium suffer from multipath fading and noise so, we need received data as receiver can be expressed as follow in time domain

$$r_k = s_k \otimes h_k + n_k$$

In which

r_k: represent the received data.

s_k: represent the sent data.

h_k: represent the channel effect

n_k: represent the AWGN

To receive the data correctly we should compensate to this channel effect. So, we need to know it as correct as possible in order to cancel it from the received data.

First the channel estimation tries to estimate the channel effect and pass these estimates to the channel equalizer. The equalizer will cancel the effect by dividing each symbol by the corresponding channel estimated value

$$y_k = s_k \times \frac{h_k}{h_{eq}} + \frac{n_k}{h_{eq}}$$

In which

h_{eq}: estimated value of the channel

In the equalizer we divide the symbol by the channel effect that was estimated by the channel estimation.

We get the symbols from resource demapper and divide them by the estimates one by one. So, the block is based on complex division.

2.9.1. Complex division

It is possible to use a complex divider directly but it is not common and not a good practice as it consumes area and power. So, we replace it with complex multipliers.

Consider $a + ib$ and $c + id$ so dividing them gives $\frac{a+ib}{c+id}$ then multiply numerator and denominator with conjugate.

$$\frac{a + ib}{c + id} \times \frac{c - id}{c - id} = \frac{(ac + bd) + i(bc - ad)}{c^2 + d^2}$$

The numerator is output of a complex multiplier and the denominator is a real positive scaling value which won’t affect the accuracy of demodulating a QPSK symbol as we concerned only with phase not magnitude.

2.10. Parallel to Serial and NRS removal

After equalization and till the end of the chain the data should be serial and the NRS are useless after channel estimation as it is not actual data and the real data at the transmitter doesn’t contain these NRS. So, the data must be converted from parallel to serial and the NRS should be removed.

2.11. Fine Synchronizer

2.11.1. Problem Definition

Assuming channel impulse response is not changed over a subframe, continuous tracking of the residual frequency offset takes place for every received subframe after successive acquisition is done at the beginning in the Coarse Synchronizer. This offset is used to correct the received symbols in carrier frequency offset block at the beginning of our chain.

The residual frequency offset extraction is done by tracking the frequency offset effects on the Narrowband Reference Signals (NRS) of different OFDM symbols, which are sent over 4 different subcarriers in the two slots of each subframe.

2.11.2. Narrowband Reference Signals NRS

They are reference signals that are used as pilots to estimate the channel effects on the received data. They are distributed within the subframe at four different subcarriers with frequency spacing of 45 kHz between each two consecutive subcarriers carrying NRS signals.

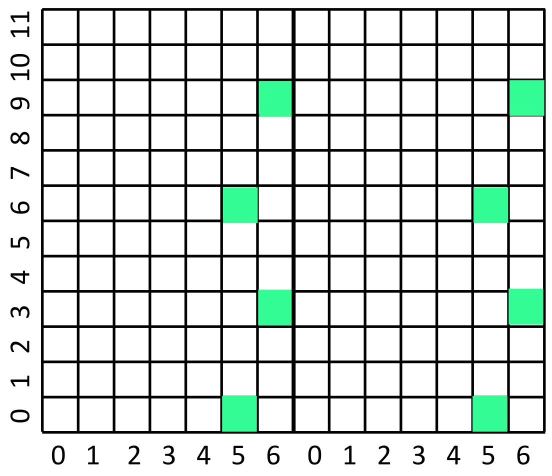


Figure [24]: Frame Pilot Locations

2.11.3. Frequency Offset Estimation

By using NRS signals of the same subcarrier we can estimate the frequency offset as we have 4 pairs of NRS signals in each subframe,

The ratio between the complex multiplication of two NRS signals of the received subframe and the locally generated NRS signals which are supposed to be received is given by the following equation

$$\frac{R_l(p)R^*_{l+N_s}(p)}{X_l(p)X^*_{l+N_s}(p)} = K\, e^{\frac{-i2\pi N_s(N+N_{CP})\varepsilon_r}{N}}$$

Where:

$X_l(p)$: Generated NRS OFDM symbol

$R_l(p)$: Received NRS ODFM symbol

N_s : Number of OFDM symbols in one slot, $N_s = 7$ in NB-IoT

p : The subcarrier frequency index

l : The OFDM time symbol index

N : Number of samples in subframe

N_{CP} : Length of Cyclic Prefix in number of samples

ε_r : Residual frequency offset.

Repeating this operation for each pair of NRS signals at each given subcarrier. The phase of the result is proportional to the required frequency offset so the frequency offset is given by:

$$\varepsilon_r = \frac{N}{2\pi N_s(N + N_{CP})} \angle\left(\sum_{p\in\upsilon} \frac{R_l(p)R^*_{l+N_s}(p)}{X_l(p)X^*_{l+N_s}(p)}\right)$$

Where:

υ : set of the four subcarriers where NRS are sent in subframe.

The phase of this residual frequency offset is desired to be calculated in order to be used in the carrier frequency offset block to eliminate this offset by correcting the received symbols.

2.12. Demodulator

2.12.1. Problem Definition

The stated demodulation scheme for the NPDSCH is the Quadrature Phase Shift keying (QPSK) which changes the carrier phase based on the input data bits. So, the demodulator reverses this operation.

QPSK modulation Scheme maps each 2 bits of data to a specific QPSK symbol having In-Phase and quadrature components as shown in the following table.

Table [8]: In phase and Quadrature components of QPSK

b_i, b_{i+1}	I	Q
00	$\frac{1}{\sqrt{2}}$	$\frac{1}{\sqrt{2}}$
01	$\frac{1}{\sqrt{2}}$	$-\frac{1}{\sqrt{2}}$
10	$-\frac{1}{\sqrt{2}}$	$\frac{1}{\sqrt{2}}$
11	$-\frac{1}{\sqrt{2}}$	$-\frac{1}{\sqrt{2}}$

So, the Demodulator in the same manner receives the QPSK symbol after equalizing the channel effects and processes them.

Using Hard Decision, the data bits are determined based on the signs of the received I and Q components in the QPSK constellation, this helps using hard decision in the rest of the receiving chain as well.

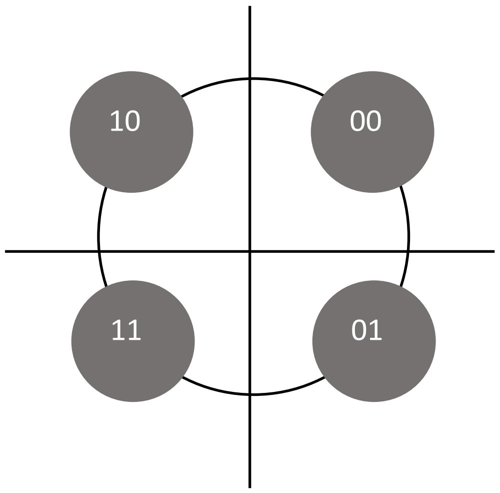


Figure [25]: QPSK Constellation Diagram

2.13. Descrambler

2.13.1. Problem Definition

Scrambling is important as it is used to eliminate long sequences of 0's and 1's by conditionally inverting some bits based on a Pseudo-random Noise sequence generated from two Linear Feedback Shift Registers and also helps in security of received data as it depends on specific parameters such as the cell identifier, and the Radio Network Temporary Identifier.

2.13.2. Polynomials and Equations

Descrambler is implemented in the same manner as the scrambler because they use the same polynomial shown down below:

$$C(n) = \big(x_1(n + N_c) + x_2(n + N_C)\big)mod2$$

And sequences $x_1$and x_2 defined as follow

$$x_1(n + 31) = \big(x_1(n + 3) + x_1(n)\big)mod2$$

Initialized with $x_1(0) = 1, x_1(n) = 0, n = 1,2, \ldots ,30$

$$x_2(n + 31) = \big(x_2(n + 3) + x_2(n + 2) + x_2(n + 1) + x_2(n)\big)mod2$$

Initialized with sequence c_{init}

$$c_{\mathrm{init}} = n_{\mathrm{RNTI}} \cdot 2^{14} + n_f \; mod \; 2 \cdot 2^{13} + \lfloor n_s/2 \rfloor \cdot 2^9 + N_{\mathrm{ID}}^{\mathrm{Ncell}} \text{ for NPDSCH}$$

Where:

$N_c = 1600$

$N_{ID}^{cell} \in (0,503)$ Cell Identifier Number and it's input from upper layer.

n_f System frame number

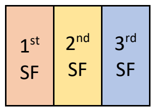
$n_s \in (0,19)$ Slot number

n_{RNTI} Radio Network Temporary Identifier

Initialization takes 1600 cycle then the sequence should stop until the input is valid then with each cycle the PN sequence is XORed with the input.

2.13.3. Descrambler Re-Initialization

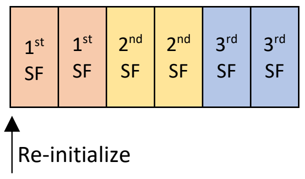
The Linear feedback shift registers get re-initialized every $\min(N_{rep}, 4)$ transmissions of the codeword. Repetitions in NPDSCH is done by repeating the message. The generated codeword in N_{SF} subframes are repeated N_{Rep} times.



A NPDSCH codeword that contains 3 subframes

DCI format type N1 parameter $I_{SF} = 2$, $N_{SF} = 3$

Case: $N_{Rep} = 2$



Case: $N_{Rep} = 8$

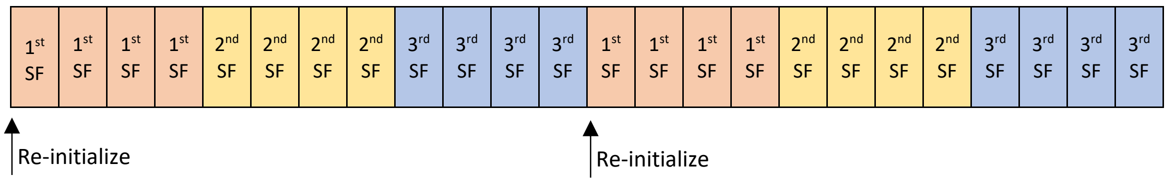


Figure [26]: Descrambler Re-initialization

2.14. Rate De-Matcher

2.14.1. Problem Definition

The rate matcher is used to improve the system performance as it sends 3 streams of the encoded data and interleaves them by a specified inter-column permutation matrix which changes the code rate of the transmitted data and distributes the information so that they don't face the same channel effects to grantee diversity.

Repetitions are handled by this block as the same codeword length can be transmitted multiple times to decrease the possibility of failure and re-transmissions.

Maximum voting technique is used to handle repetitions in the bit collection stage.

2.14.2. The Rate matcher

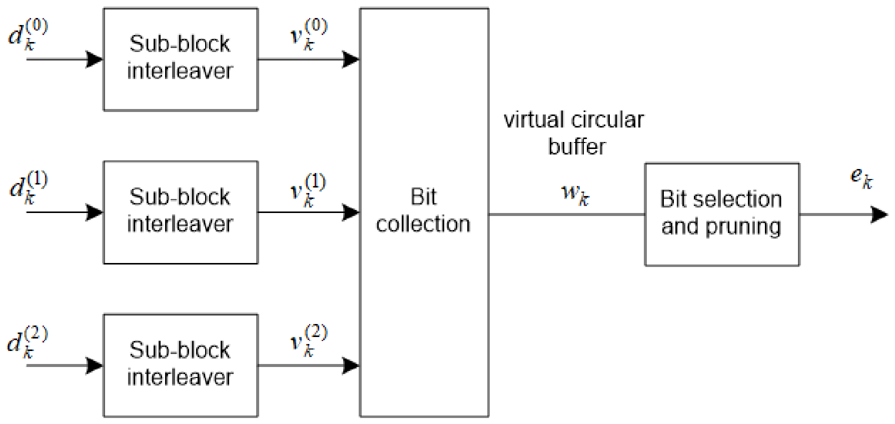


Figure [27]: Rate Matcher Subblocks

The input to the Rate matcher block is the 3 outputs of the tail-biting convolutional encoder $d_k^{(1)}, d_k^{(2)}, d_k^{(3)}$ each of length D as D is the number of bits

2.14.2.1. Sub-block interleaver

A reshape matrix of number of columns equal 32 and number of rows equal the ceiling of division of number of bits by 32, then dummy bits of number N_D are added to the beginning of this matrix such that

$$C_{subblock}^{CC} = 32$$

$$R_{subblock}^{CC} = \left\lceil \frac{D}{C_{subblock}^{CC}} \right\rceil$$

$$K_{\pi} = C_{subblock}^{CC} x R_{subblock}^{CC}$$

$$N_D = K_{\pi} - D$$

And the interleavers are filled by y_k where:

$$y_k = < NULL > \text{ for k = 0,1, …, } N_D$$

$$y_{N_D+k} = d_k^{(1)} \quad \text{for k = 0,1, …, D}$$

The three encoded data streams are stored in three matricides by filling them row by row using the incoming data after the first N_D bits which are dummy bits.

$$\begin{bmatrix} y_0 & y_1 & y_2 & \cdots & y_{C_{subblock}^{CC}-1} \\ y_{C_{subblock}^{CC}} & y_{C_{subblock}^{CC}+1} & y_{C_{subblock}^{CC}+2} & \cdots & y_{2C_{subblock}^{CC}-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{(R_{subblock}^{CC}-1)\times C_{subblock}^{CC}} & y_{(R_{subblock}^{CC}-1)\times C_{subblock}^{CC}+1} & y_{(R_{subblock}^{CC}-1)\times C_{subblock}^{CC}+2} & \cdots & y_{(R_{subblock}^{CC}\times C_{subblock}^{CC}-1)} \end{bmatrix}$$

Figure [28]: Interleaver Matrix

The Encoded bits of each bit stream gets interleaved based on an intercolumn permutation pattern shown:

Number of columns $C_{subblock}^{CC}$	**Inter-column permutation pattern** $< P(0), P(1), \ldots, P(C_{subblock}^{CC}-1) >$
32	< 1, 17, 9, 25, 5, 21, 13, 29, 3, 19, 11, 27, 7, 23, 15, 31, 0, 16, 8, 24, 4, 20, 12, 28, 2, 18, 10, 26, 6, 22, 14, 30 >

Figure [29]: Interleavers Intercolumn Permutation Pattern

Then after performing the inter-column permutation the result is

$$\begin{bmatrix} y_{P(0)} & y_{P(1)} & y_{P(2)} & \cdots & y_{P(C_{subblock}^{CC}-1)} \\ y_{P(0)+C_{subblock}^{CC}} & y_{P(1)+C_{subblock}^{CC}} & y_{P(2)+C_{subblock}^{CC}} & \cdots & y_{P(C_{subblock}^{CC}-1)+C_{subblock}^{CC}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{P(0)+(R_{subblock}^{CC}-1)\times C_{subblock}^{CC}} & y_{P(1)+(R_{subblock}^{CC}-1)\times C_{subblock}^{CC}} & y_{P(2)+(R_{subblock}^{CC}-1)\times C_{subblock}^{CC}} & \cdots & y_{P(C_{subblock}^{CC}-1)+(R_{subblock}^{CC}-1)\times C_{subblock}^{CC}} \end{bmatrix}$$

Figure [30]: Interleaver Matrix after performing intercolumn change

2.14.2.2. Bit Collection and transmission

The output of the block interleaver is read column by column and fill the circular buffer of length $K_w = 3K_\pi$ with the data of each interleaver in the following manner,

$$w_k = v_k^{(1)}$$

$$w_{K_\pi+k} = v_k^{(2)}$$

$$w_{2K_\pi+k} = v_k^{(3)}$$

After that, data is read from the circular buffer by eliminating the dummy bits until reach E

where E is the output sequence length

The following Pseudo code represents how the output is generated.

output= e_k where k = 0,1, …, E-1

```
Set $k = 0$ and $j = 0$
while {$k < E$}
    if $w_{j \bmod K_w} \neq <NULL>$
        $e_k = w_{j \bmod K_w}$
        $k = k + 1$
    end if
    $j = j + 1$
end while
```

2.14.3. The Rate De-matcher:

2.14.3.1. Bit collection

Bit collecting which is a memory of size that should cover the length of the circular buffer for the maximum TBS that can be transmit which is 2536 and the maximum repetitions number which is 2048 in release 14.

$$Circular\ buffer\ length = 3 * (\text{Max TBS } + 24 \text{ bits CRC}) * \log_2(\max repetition\ size)$$

There are 3 possibilities for the value of the input data length E:

1. E = K_w, then the value saved in the buffer will be passed to the interleavers
2. E > K_w, then the incoming bit stream is added to the previous repetitions of data and then take the average of them and compare it to 0.5, if it is greater than or equal the value is assumed to be 1 else, it is a 0, which is called maximum voting
3. E < K_w, then the buffer is zero-padded

2.14.3.2. Sub-block De-interleaver

Interleavers which are three memories of size that should cover the length of K_π and consists of $C_{subblock}^{CC}$ columns and $R_{subblock}^{CC}$rows.

$$Interleaver\ length = C_{subblock}^{CC} * \left\lceil \frac{\text{Max TBS } + 24 \text{ bits CRC}}{C_{subblock}^{CC}} \right\rceil$$

The bit collection stage output is the input for the interleavers which are written in a specific order following the permutation pattern and then read in another order to execute the interleaving process.

Rate De-matcher de-interleavers input writing direction:

$$\Bigg\downarrow \begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} & \cdots & a_{0,31} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} & \cdots & a_{1,31} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & \cdots & a_{2,31} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{r,0} & a_{r,1} & a_{r,2} & a_{r,3} & a_{r,4} & \cdots & a_{r,31} \end{bmatrix}$$

Rate De-matcher de-interleavers output reading direction:

$$\xrightarrow{\qquad\qquad\qquad\qquad}$$

$$\begin{bmatrix} a_{0,1} & a_{0,17} & a_{0,9} & a_{0,25} & a_{0,5} & \cdots & a_{0,30} \\ a_{1,1} & a_{1,17} & a_{1,9} & a_{1,25} & a_{1,5} & \cdots & a_{1,30} \\ a_{2,1} & a_{2,17} & a_{2,9} & a_{2,25} & a_{2,5} & \cdots & a_{2,30} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{r,1} & a_{r,17} & a_{r,9} & a_{r,25} & a_{r,5} & \cdots & a_{r,30} \end{bmatrix}$$

2.15. Viterbi Decoder

2.15.1. Tail Biting Convolutional Encoder

A Tail Biting Convolutional Encoder (TBCC) with constraint length 7 and a rate of 1/3 is defined in 3GPP 36.212 standard as shown in the figure with polynomials G0 = 133, G1=171, G2=165 (octal).

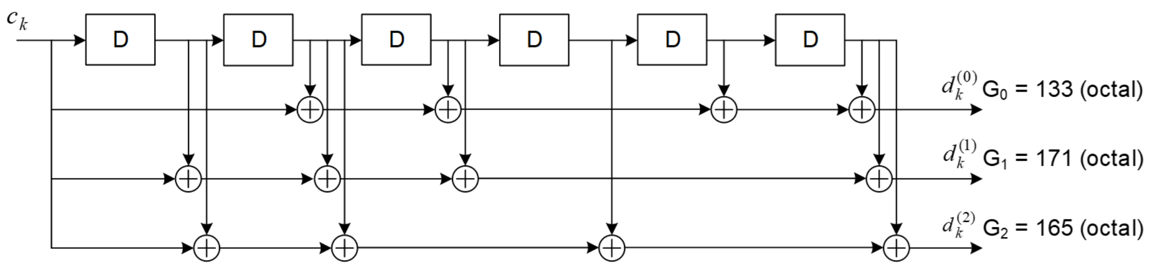

c_k
D
D
D
D
D
D
$d_k^{(0)}$ G_0 = 133 (octal)
$d_k^{(1)}$ G_1 = 171 (octal)
$d_k^{(2)}$ G_2 = 165 (octal)

Figure [31]: Tail Biting Convolutional Encoder for NB-IOT LTE

Another requirement is defined to initialize the states of encoder's shift register. The initial value of the shift register is the last 6 bits of the transmitted block. This results in having the initial state of the shift register equal to the final state of the shift register. This increases the decoder complexity but also doesn't affect the code rate.

2.15.2. Viterbi Algorithm

Viterbi algorithm is a maximum likelihood method to detect the most probable sequence of states given certain bits. It calculates the similarities (Hamming Distance in hard decision decoders) between the received bits and all trellis paths entering each state at a time as shown in the figure.

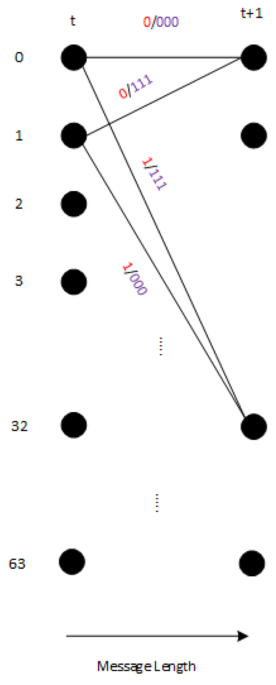


Figure [32]: Trellis Diagram

2.15.3. System Requirements

The algorithm needed for the system must have a lower complexity as we need to reduce the power, the winning path must be a tail biting path as the encoder implicitly made the initial state and the final state equal, the available states in the trellis diagram are 64 states as we have a 6-bit shift register and the code rate is 1/3 so the received message will be divided into 3-bit segments. The algorithm chosen for the system is Wrap Around Viterbi Algorithm (WAVA).

2.15.4. Wrap Around Viterbi Algorithm

It is an iterative algorithm that applies the non-fixed state Viterbi algorithm and keeps applying itself until the winning path becomes a tail biting path where the initial and final states become equal. Steps of the algorithm are as shown:

1. Start from all the state with state metrics set to 0
2. Proceed in the trellis diagram calculating Branch Metrics, Path Metrics and Survived Paths for all states.
3. Perform the traceback operation to reach the initial state to evaluate the Tailbiting Condition.
4. If the Tailbiting Condition is true, assert the valid signal and start the output stream of decoded bits.
5. If the Tailbiting Condition is false, go to step 2 and keep the current values of the path metrics saved for the next iteration.
6. If the number of iterations becomes 3, assert the valid signal and start the output stream of decoded bits.

2.16. Cyclic Redundancy Check

Cyclic redundancy check is an error detecting code used to validate received data integrity by appending a specific value to the data which is the remainder in case of dividing this data by w specific polynomial, which makes the result divisible by this polynomial so at the receiver the result of this operation should result in Zero which means the received data is correct.

The parity bits are generated by the following cyclic generator polynomial:

$$g_{CRC24A}(\mathrm{D}) = D^{24} + D^{23} + D^{18} + D^{17} + D^{14} + D^{11} + D^{10} + D^{7} + D^{6} + D^{5} + D^{4} + D^{3} + \mathrm{D} + 1$$

Where the input bits are donated by $a_0, a_1, a_2, a_3, \dots, a_{A-1}$, while the parity bits are donated by $p_0, p_1, p_2, p_3, \dots, p_{L-1}$

After appending the CRC parity bits, the output is donated by $b_0, b_1, b_2, b_3, \dots, b_{B-1}$where B=A+L

$$b_k = a_k \text{ for } k = 0, 1, 2, \dots, A\text{-}1$$

$$b_k = p_{k-A} \text{ for } k = A, A+1, A+2, \dots, A+L\text{-}1.$$

Figure [33]: CRC appending

Cyclic redundancy check is implemented in the same manner in the receiver because they use the same polynomial.

3.1. Coarse Synchronizer

3.1.1. Top Module

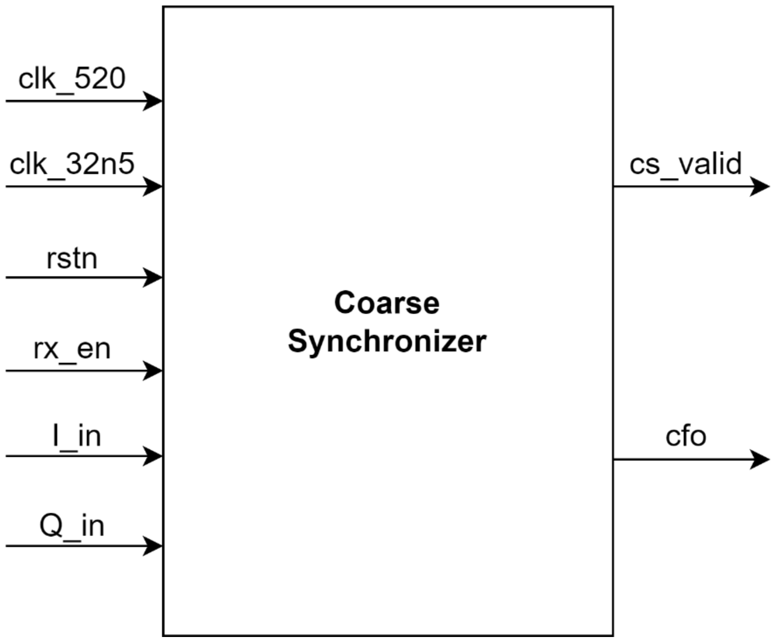


Figure [34]: Coarse Synchronizer Top Module

Table [9]: Coarse Synchronizer Interface Table

Signal Name	Direction	Width	Description
clk_520	Input	1	1.92 MHz clock signal
clk_32n5	Input	1	30.769 MHz clock signal for internal CORDIC units in Fine-Tuning Stage
rstn	Input	1	Global reset signal
rx_en	Input	1	Enable Signal that indicates the start of NB-IoT Receiver Operation
I_in	Input	16	Real serial input samples
Q_in	Input	16	Imaginary serial input samples
cs_valid	Output	1	Valid Signal indicating the validity of the output CFO
cfo	Output	19	Estimated Carrier Frequency Offset

3.1.2. Detailed Hardware

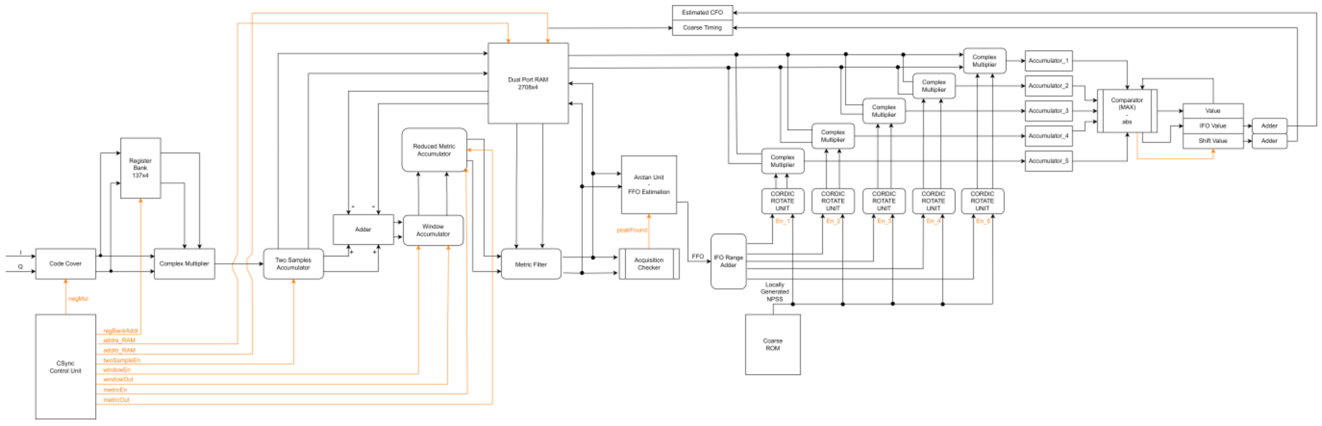


Figure [35]: Coarse Synchronizer Detailed Hardware

3.1.3. Design Challenges and Solutions

3.1.3.1. High Power Consumption

Coarse Synchronizer is the most power-hungry block in the RX chain as it is one of the largest in area and contains many arithmetic circuits for metric calculations and frame processing. In order to minimize the power consumption, we used some of the **Low Power Design Techniques** in the block design

- Operand Isolation: By inserting pipeline registers before any arithmetic operation, this is because one of the operands bits might arrive at different time than other bits within the same clock cycle, making the arithmetic circuit operate multiple times within the same clock cycle. By Registering the input operands, we ensure that the arithmetic circuit will operate only once as they are already after the active clock edge.

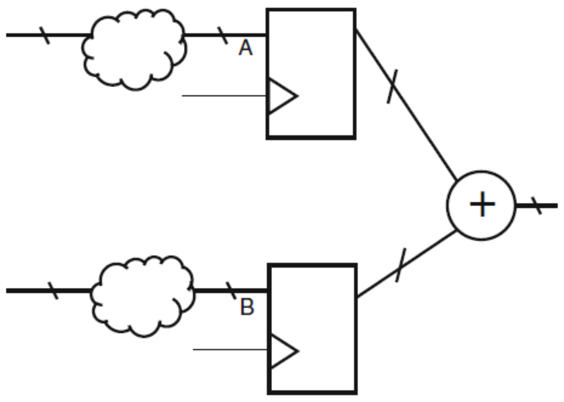


Figure [36]: LPDT-Operand Isolation

- Comparison Priority: Coarse Synchronizer continuously compares metric values with the Acquisition Threshold which consumes high power as comparator operations involve XORs which have high switching activity factor, thus, consuming more power. So, when we have to compare large data busses, we avoided comparing all the bits in one go but rather we compared the MSB's first, if it satisfies the condition, there's no need to evaluate the rest of the bit XORing. If not, then we compare the rest of the bus bits.

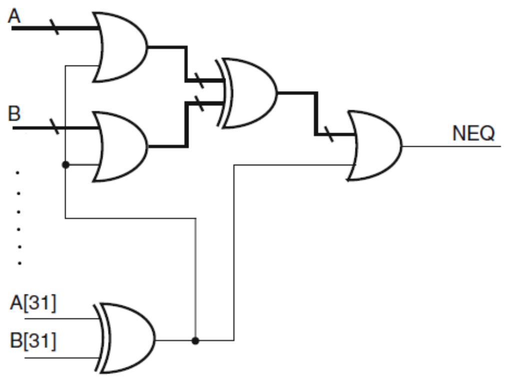


Figure [37]: LPDT-Comparison Priority

3.1.3.2. Large Memory Requirements

Coarse Synchronizer requires storing different window metric values along the NB-IoT frame which consumes around $\left(\frac{19200*32}{8*1024} = 75\ KB\right)$. We solved this problem by successive addition of every 16-window metric with each other and fine-tuning these metric accumulations in the Fine-Tuning Stage to preserve the CFO extraction accuracy. Resulting in the need of only $\frac{1200*32}{8*1024} = 4.6875\ KB$.

3.1.3.3. Window's Metric Huge Latency

Window sliding to calculate coarse timing requires processing on 1508 samples in every window over the entire NB-IoT frame which introduces huge latency in the block. Thus, we decided to store the window sample multiplications so that this processing occurs only once for the first window, and every new sample, a new window is evaluated by subtracting the multiplication of first samples and adding the multiplication of the newer samples. Thus, decreasing block's latency.

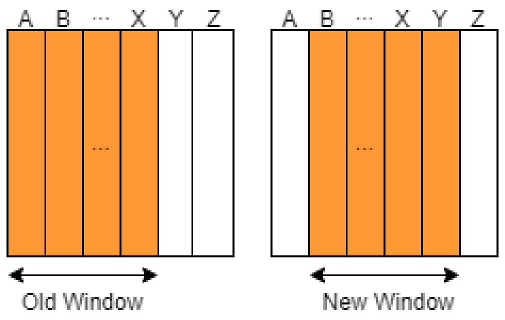


Figure [38]: Window Sliding

3.1.4. MATLAB Results

MathsWorks LTE Toolbox were used to generate input vectors to the Coarse Synchronizer MATLAB model for algorithm's accuracy testing.

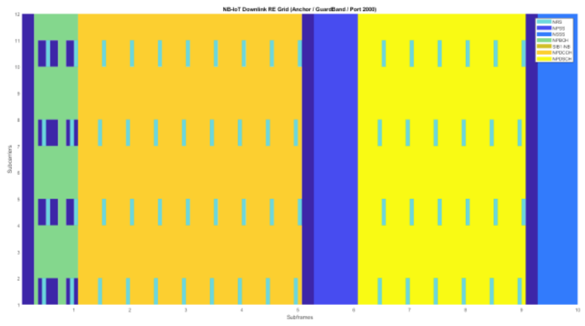
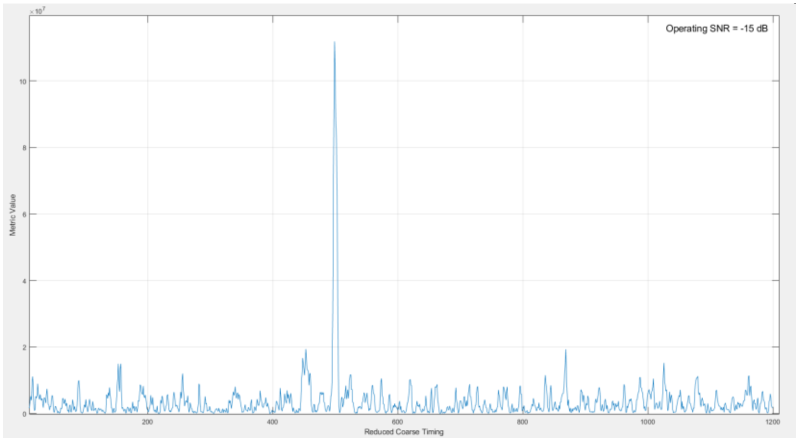
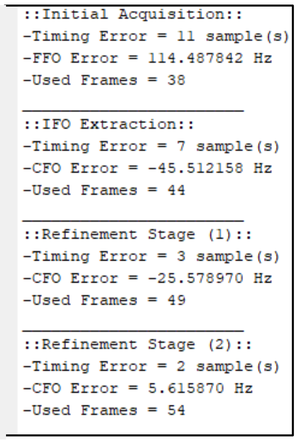


Figure [39]: NB-IoT One Frame RE Grid

Operating on randomly generated data on $SNR = -15\ dB$, the figure below shows the acquisition results along with estimated coarse timing and CFO extraction errors.



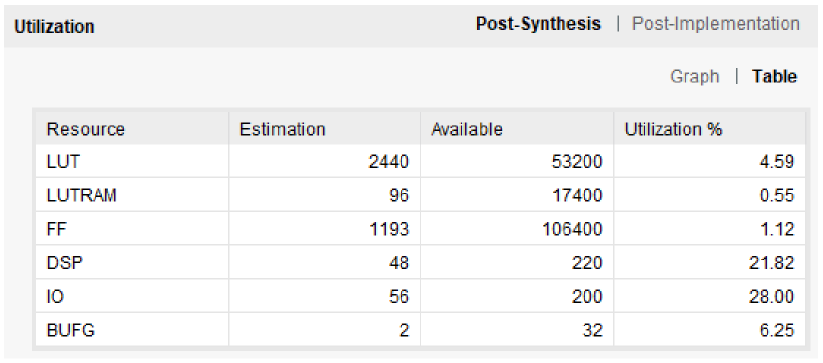


```
::Initial Acquisition::
-Timing Error = 11 sample(s)
-FFO Error = 114.487842 Hz
-Used Frames = 38
________________________________
::IFO Extraction::
-Timing Error = 7 sample(s)
-CFO Error = -45.512158 Hz
-Used Frames = 44
________________________________
::Refinement Stage (1)::
-Timing Error = 3 sample(s)
-CFO Error = -25.578970 Hz
-Used Frames = 49
________________________________
::Refinement Stage (2)::
-Timing Error = 2 sample(s)
-CFO Error = 5.615870 Hz
-Used Frames = 54
```

Figure [40]: Acquisition and CFO Extraction MATLAB Results

3.1.5. Synthesis Results

The Figure below shows the overall synthesized area in FPGA cells on the targeted FPGA. Synthesis is done using Vivado Design Suite which includes block's utilization in resources (LUTs, FFs, DSPs, etc.)


Utilization | Post-Synthesis | Post-Implementation

Graph | Table

Resource	Estimation	Available	Utilization %
LUT	2440	53200	4.59
LUTRAM	96	17400	0.55
FF	1193	106400	1.12
DSP	48	220	21.82
IO	56	200	28.00
BUFG	2	32	6.25

Figure [41]: Coarse Synchronizer Synthesis Utilization

3.2. CP Remover and Downsampler

3.2.1. Top Module

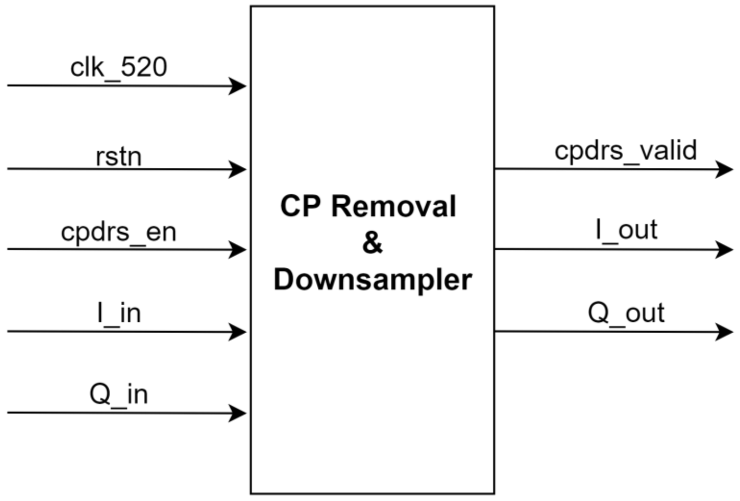
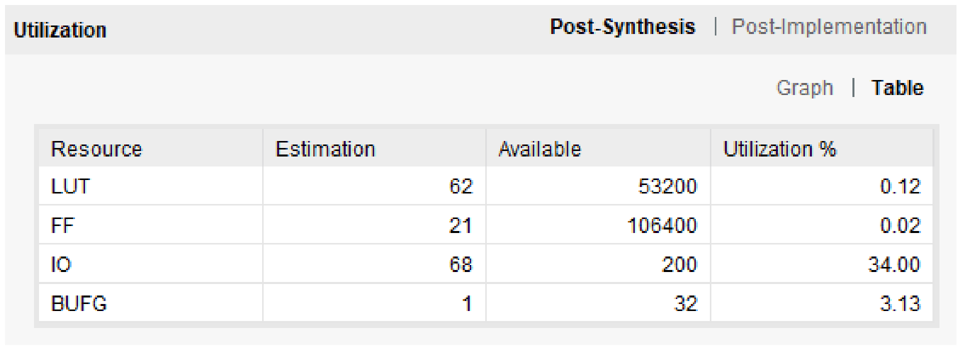


Figure [42]: CP Remover and Downsampler Top Module

Table [10]: CP Remover and Downsampler Interface Table

Signal Name	Direction	Width	Description
clk_520	Input	1	1.92 MHz clock signal
rstn	Input	1	Global reset signal
cprds_en	Input	1	Enable Signal that indicates the start block's operation
I_in	Input	16	Real serial input samples
Q_in	Input	16	Imaginary serial input samples
cprds_valid	Output	1	Valid Signal indicating the validity of the output samples
I_out	Output	16	Real serial output samples
Q_out	Output	16	Imaginary serial output samples

3.2.2. Synthesis Results



Resource	Estimation	Available	Utilization %
LUT	62	53200	0.12
FF	21	106400	0.02
IO	68	200	34.00
BUFG	1	32	3.13

Figure [43]: CPRDS Synthesis Utilization

3.3. CFO Corrector

3.3.1. Top Module

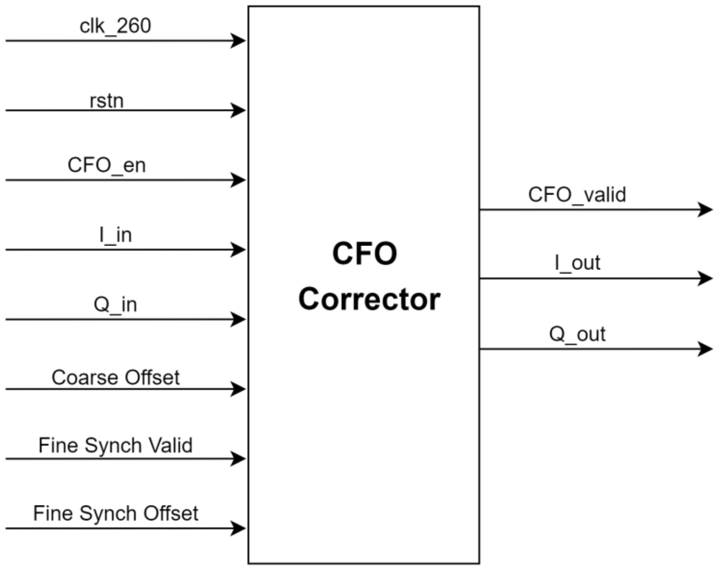


Figure [44]: CFO Corrector Top Module

Table [11]: CFO Corrector Interface Table

Signal Name	Direction	Width	Description
clk_260	Input	1	3.84 MHz clock signal
rstn	Input	1	Global reset signal
CFO_en	Input	1	Enable Signal that indicates the start block's operation
I_in	Input	16	Real serial input samples
Q_in	Input	16	Imaginary serial input samples
Coarse Offset	Input	19	Estimated Carrier Frequency Offset from Coarse Synchronizer

Fine Synch Valid	Input	1	Valid Signal indicating the validity of Fine Synchronizer offset
Fine Synch Offset	Input	19	Estimated Fine Synchronizer Offset
CFO_valid	Output	1	Valid Signal indicating the validity of the output samples after correction
I_out	Output	16	Real serial output samples
Q_out	Output	16	Imaginary serial output samples

3.3.2. Detailed Hardware

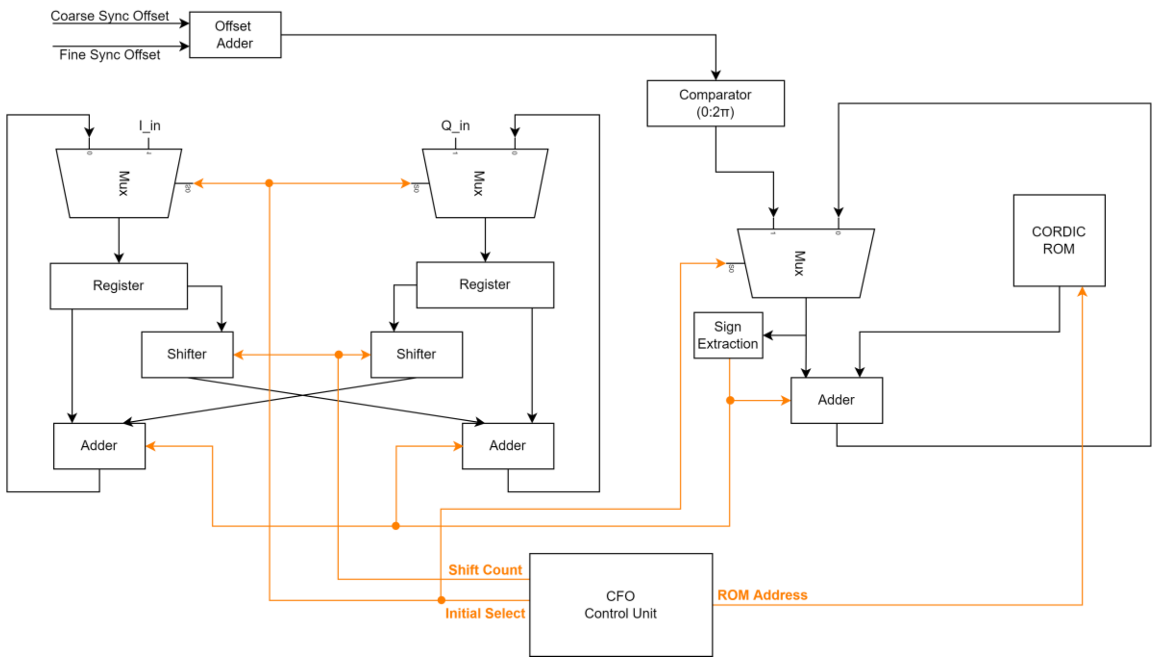


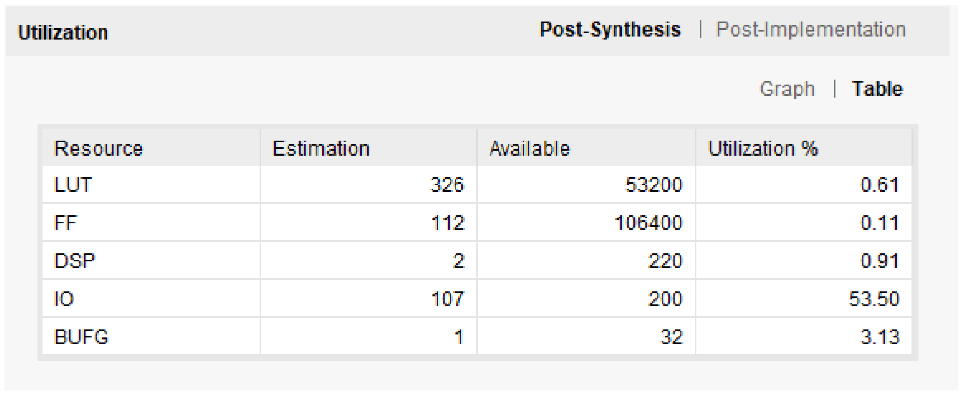
Figure [45]: CFO Corrector Detailed Hardware

3.3.3. Design Challenges and Solutions

3.3.3.1. Area Utilization

There is a trade-off between CFO Corrector's area and latency. Expanded version of a CORDIC Unit could be implemented by using 15 parallel stages and output could be obtained in one clock cycle. Nevertheless, this very low latency will be redundant as CFO Corrector samples a new input every 8 clock cycles coming from Downsampler. So, we decided to go with an iterative implementation by only using one stage and reusing the same hardware for all CORDIC steps and operating at double the frequency. Resulting in no wasted clock cycles and 93.5% improvement in area.

3.3.4. Synthesis Results


Utilization | **Post-Synthesis** | Post-Implementation

Graph | **Table**

Resource	Estimation	Available	Utilization %
LUT	326	53200	0.61
FF	112	106400	0.11
DSP	2	220	0.91
IO	107	200	53.50
BUFG	1	32	3.13

Figure [46]: CFO Corrector Synthesis Utilization

3.4. FFT Engine

3.4.1. Top Module

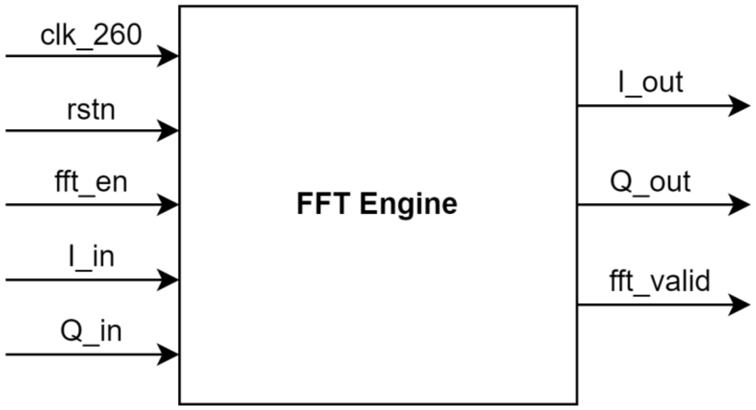


Figure [47]: FFT Top Module

Table [12]: FFT Engine Interface Table

Signal Name	Direction	Width	Description
clk_260	Input	1	3.84 MHz clock signal
rstn	Input	1	Global reset signal
cprds_en	Input	1	Enable Signal that indicates the start block's operation
I_in	Input	16	Real serial input samples
Q_in	Input	16	Imaginary serial input samples
cprds_valid	Output	1	Valid Signal indicating the validity of the output samples
I_out	Output	16	Real serial output samples
Q_out	Output	16	Imaginary serial output samples

FFT Specifications:

- 16-point processing
- Discrimination in Frequency (DIF)
- SDF Architecture
- Radix 2^2 Algorithm

Data flow through stages:

At first, 16-time domain samples coming from CFO Corrector are stored in 8×4 memory and to improve system latency the first stage will start to operate as soon as the 9th sample is stored in the 1st stage's memory. Output of first stage will propagate to 2nd stage, and like in 1st stage, 2nd stage will start operating as soon as the 5th output from stage 1 arrives. This will also happen for 3rd and 4th stage until QPSK symbols begin to show at the output serially one by one. Block's latency is 16 clock cycle, once the pipeline is filled, a QPSK symbol will be ready every one clock cycle.

3.4.2. Detailed Hardware

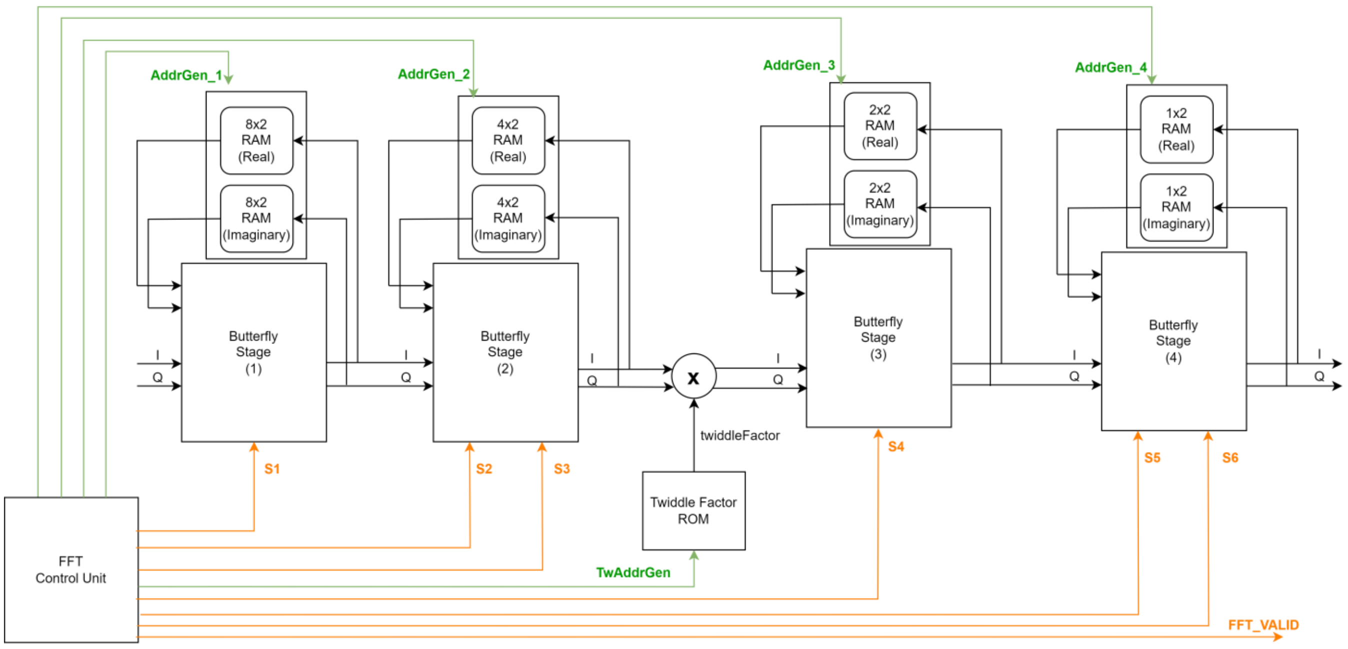


Figure [48]: FFT Engine Detailed Hardware

Regarding DIF architectures, data widths increase linearly as delay-line memory depths decrease exponentially. This means that restraining bit growth in DIF FFT processors results in minimal savings as compared to the potential impacts of quantization. Memories used to implement delay-lines for SDF FFT processors do not require random access. A straightforward sequential access scheme in which read and write pointers are simultaneously incremented for each pair of complex data samples for all four stages.

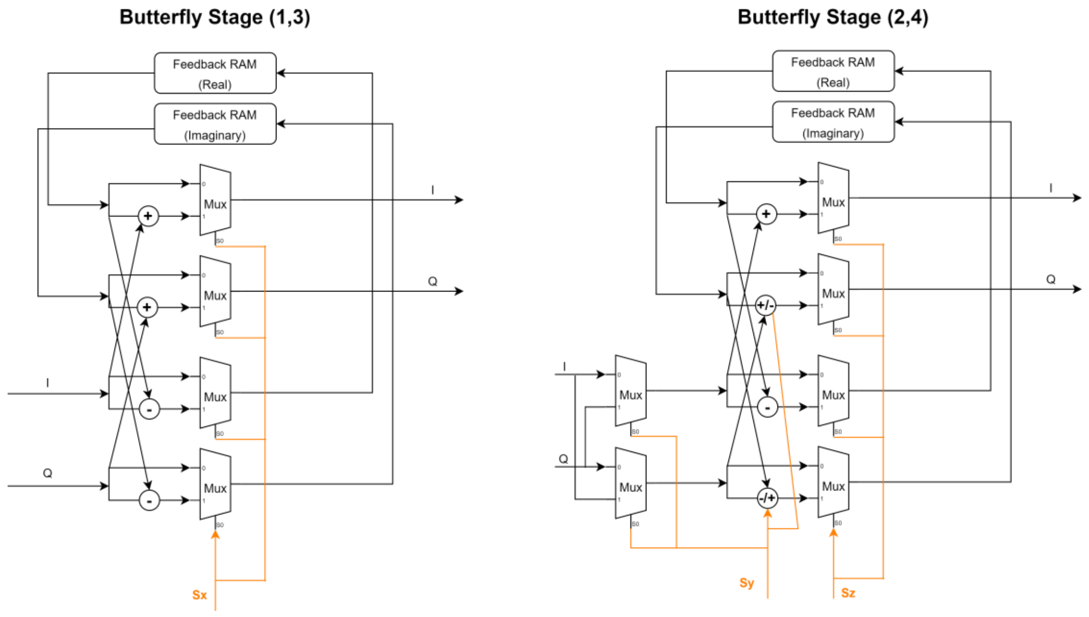
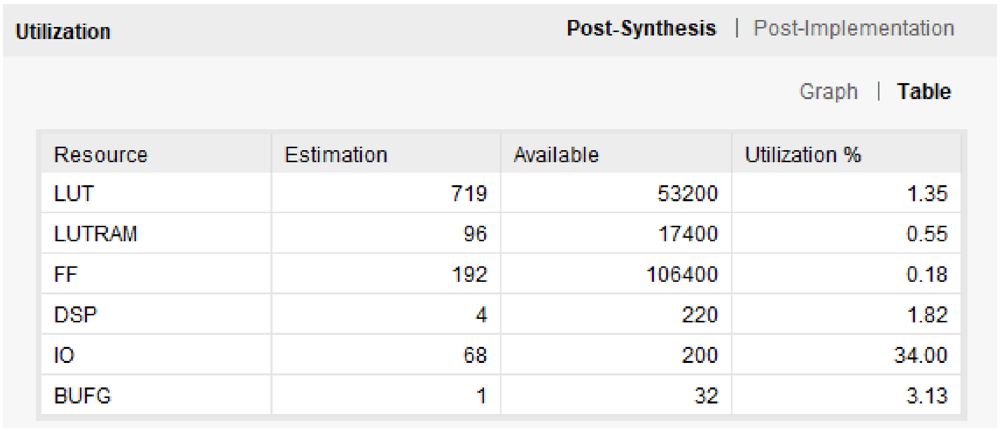


Figure [49]: Butterfly Hardware Structure

As shown in butterfly hardware structure, BF (1,3) and BF (2,4) are identical except that BF (2,4) have some added logic to perform a ±j multiplication without the need of a complex multiplier. This is done by adding a multiplexing level to swap between Real and Imaginary stage inputs, and a control signal coming from control unit to invert the sign in the butterfly adders.

In the Radix-2^2 SDF architecture, a twiddle multiplication stage is implemented after every two butterfly stages. At every twiddle stage, a complex hardware multiplier is used to multiply each data sample by a corresponding complex twiddle coefficient of unit magnitude. The product is then truncated down to the bit width of the data stream before entering the subsequent butterfly stage.

3.4.3. Synthesis Results



Resource	Estimation	Available	Utilization %
LUT	719	53200	1.35
LUTRAM	96	17400	0.55
FF	192	106400	0.18
DSP	4	220	1.82
IO	68	200	34.00
BUFG	1	32	3.13

Figure [50]: FFT Synthesis Utilization

3.5. Resource Demapper

3.5.1. Top Module

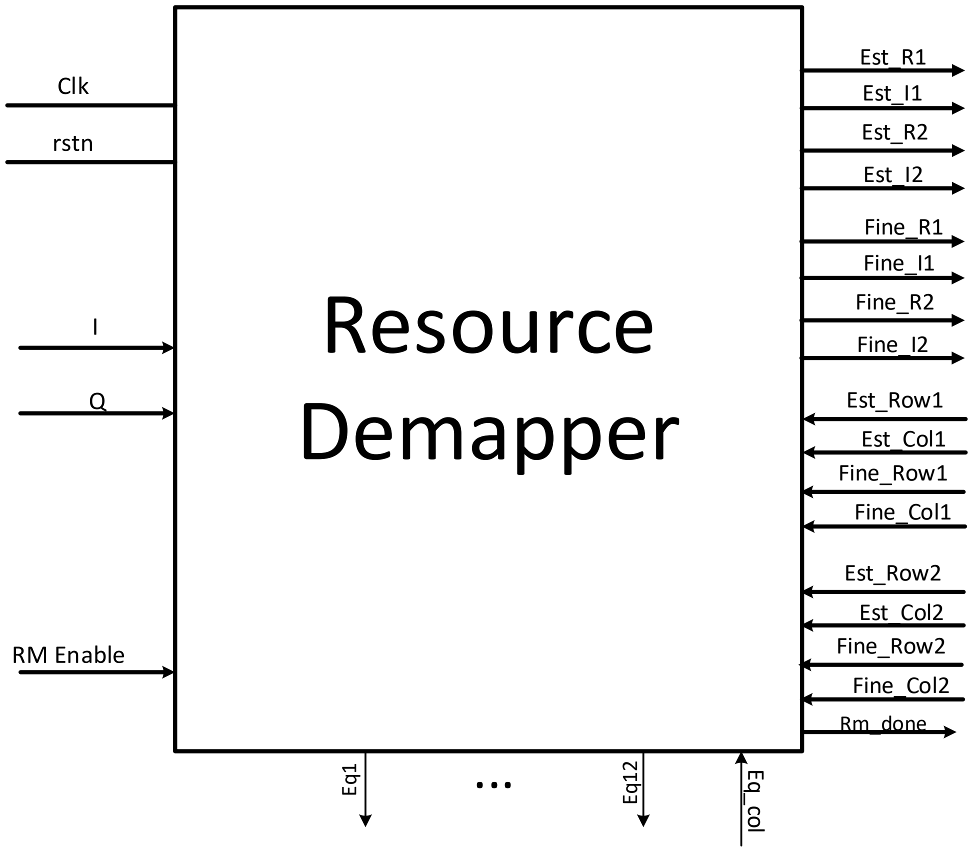


Figure [51]: Resource Demapper Top Module

Table [13]: Resource Demapper Interface Table

Signal Name	Direction	Width	Description
clk	Input	1	260ns clock signal
Rstn	Input	1	Active low reset
I	Input	16	Real part of received sample from fft
Q	Input	16	Imaginary part of received sample from fft
rmEnable	Input	1	Block enable from the fft
Eq_col	Input	4	Number of the column to be equalized
Est_row1	Input	4	The row of the needed pilot by the estimator
Est_col1	Input	4	The column of the needed pilot by the estimator
Est_row2	Input	4	The row of the needed pilot by the estimator
Est_col2	Input	4	The column of the needed pilot by the estimator
Fine_row1	Input	4	The row of the needed pilot by fine synchronizer
Fine_col1	Input	4	The column of the needed pilot by fine synchronizer
Fine_row2	Input	4	The row of the needed pilot by fine synchronizer
Fine_col2	Input	4	The column of the needed pilot by fine synchronizer

Est_R1	Output	16	Received pilot real part
Est_I1	Output	16	Received pilot imaginary part
Est_R2	Output	16	Received pilot real part
Est_I2	Output	16	Received pilot imaginary part
fine_R1	Output	16	Received pilot real part
fine_I1	Output	16	Received pilot imaginary part
fine_R2	Output	16	Received pilot real part
fine_I2	Output	16	Received pilot imaginary part
Eq1 to Eq12 R	Output	16	Received symbol’s real part to be equalized
Eq1 to Eq12 I	Output	16	Received symbol’s imaginary part to be equalized
Rm_done	Output	1	Signal indicates that storing is done

3.5.2. Detailed Hardware

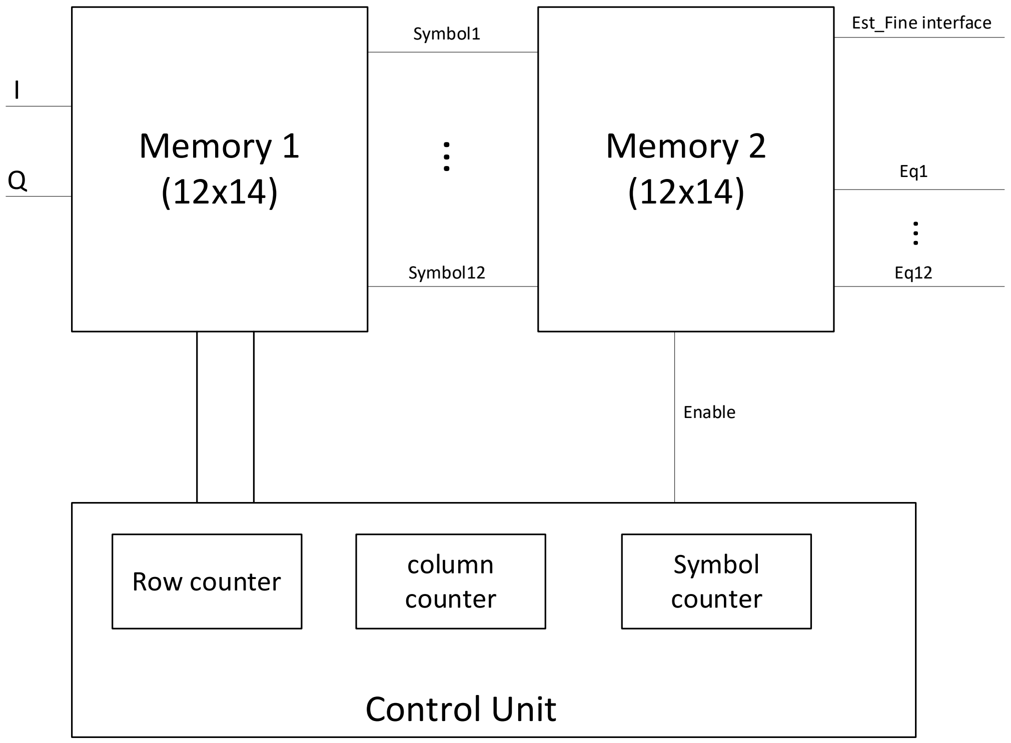


Figure [52]: Resource demapper detailed hardware

The block consists of two memories 12x14 each to store the whole subframe which consists of 12 subcarriers and 14 OFDM symbol, the 1st memory takes the output of the FFT sample by sample (QPSK symbol) constructing an OFDM symbol every 16 cycles then after a complete OFDM symbol we start to fill the second column in the memory.

The row and column counter are used to keep tracking how many places are filled and also used as addresses for the data in the first memory then the symbol counter is used to count the number of OFDM symbol transferred from the 1st memory to the 2nd memory.

After storing the whole subframe (14 OFDM symbol) the 1st memory raises the write enable of the 2nd memory and transfer to it symbol by symbol till the second memory is full and store the whole subframe, while the 2nd memory stores the subframe the 1st memory will keep storing in the new subframe.

By this we guarantee the stability of data in the memory within our processing time

3.5.3. Design Challenges and Solutions

3.5.3.1. Timing

We need to know the processing time which is the time that we need the data to be constant in the memory, so this time determine our memory width.

$$time = (NRS\ generation\ latency)(clk) + (channel\ estimation\ latency\ )(clk) + (equalizater\ latency\ )(clk)$$

$$time = 6424 \times 130ns + 7 \times 520ns + 1 \times 520ns \cong 0.85ms$$

Knowing that the subframe time is around $0.95ms$ so we need to keep the whole subframe constant using the 12×14 memories

3.5.3.2. Removing Dummy

The data output from our FFT has 4 dummy samples. The dummy is inserted at the Tx because we use 16-point FFT while the OFDM symbol contain only 12 sample so 4 dummy samples are added to work with the 16-point FFT.

There was a solution of storing the whole received samples letting the 1st memory to be 16x14 memory which we considered as high overhead because this gives 4 dummy samples in 14 symbol each of 16bit real and 16 imaginary and this is a total of $4 * 14 * 16 * 2 = 1792\ bit$.

The dummy samples output from the fft will not be stored as we know when they are coming as they are at well-known places so we know when they will come and disable the write enable of the memory so as not to store them.

3.5.3.3. Ordering

The data output from the FFT is out of order as the FFT uses DIF so its input is in order but the output is out of order.

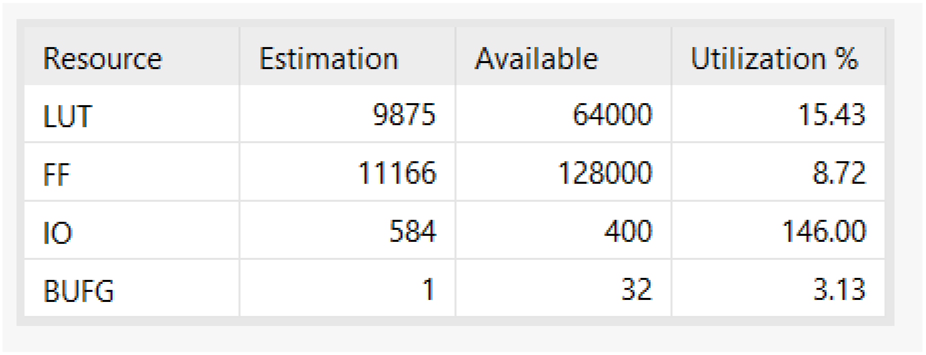
Instead of using complex control to control the storing in different order than the order the data comes in with we decided to normally store the data in the 1st memory then while transferring from the 1st memory to the 2nd memory we use wiring to do the needed ordering. By that the data in the 2nd memory are in the required normal order without adding complexity of control.

Table [14]: Ordering table

Input	Stored order
0	0
1	8
2	4
3	2
4	6
5	1
6	9
7	13
8	3
9	11
10	7
11	15

3.5.4. Results

3.5.4.1. Synthesis Result



Resource	Estimation	Available	Utilization %
LUT	9875	64000	15.43
FF	11166	128000	8.72
IO	584	400	146.00
BUFG	1	32	3.13

Figure [53]: Resource Demapper Synthesis result

3.6. Channel Estimation

3.6.1. Top Module

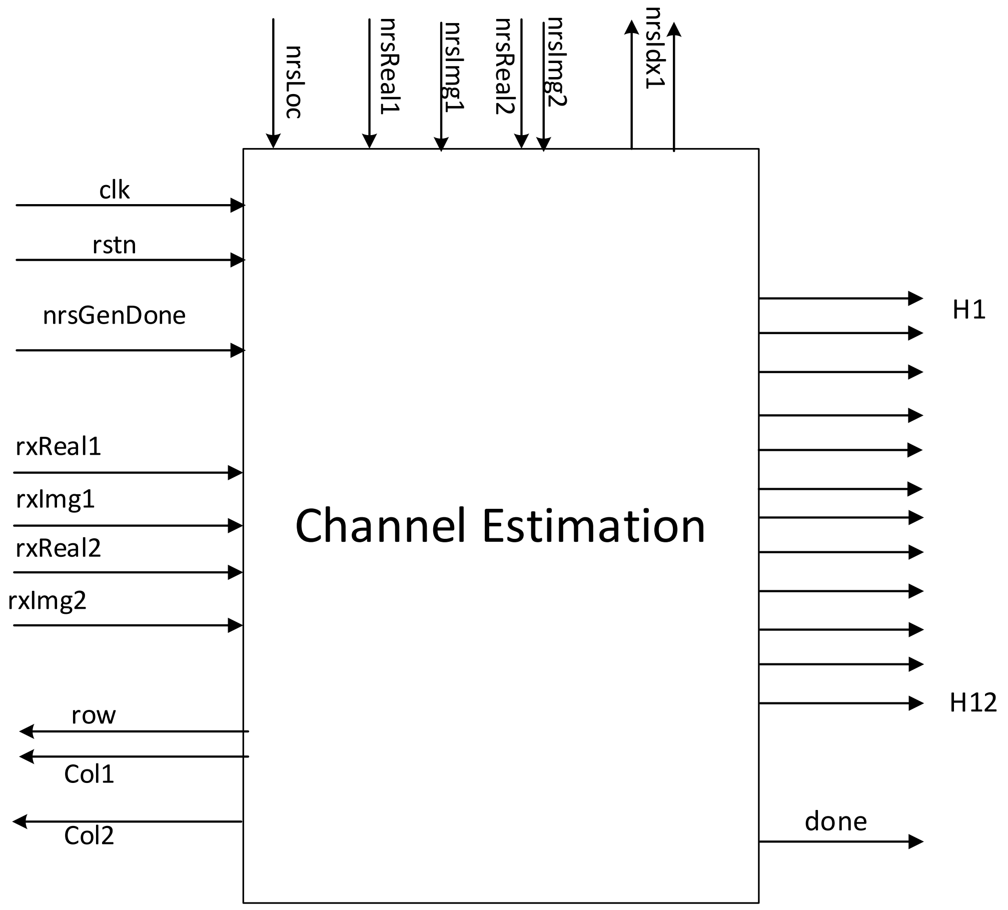


Figure [54]: Channel Estimation Top Module

Table [15]: Channel Estimation Interface Table

Signal Name	Direction	Width	Description
Clk	Input	1	520ns clock signal
Rstn	Input	1	Active low reset signal
nrsGenDone	Input	1	Signal indicate that pilot generation has finished
rxReal1	Input	16	Real part of the received pilot
rxImg1	Input	16	Imaginary part of the received pilot
rxReal2	Input	16	Real part of the received pilot
rxImg2	Input	16	Imaginary part of the received pilot
nrsLoc	Input	4	Index that we get from the NRS index generator
nrsReal1	Input	16	Real part of locally generated pilot
nrsImg1	Input	16	Imaginary part of locally generated pilot
nrsReal2	Input	16	Real part of locally generated pilot
nrsImg2	Input	16	Imaginary part of locally generated pilot

nrsIdx1	Output	3	Index used to access NRS index generator memory
nrsIdx2	Output	3	Index used to access NRS index generator memory
Row	Output	4	Row of the received pilot in the resource block
Col1	Output	4	Column of the received pilot in the resource block
Col2	Output	4	Column of the received pilot in the resource block
H1 … H12 Real	Output	16	Estimated real part
H1 … H12 Img	Output	16	Estimated imaginary part
done	Output	1	Signal verifies that estimation is done

3.6.2. Detailed Hardware

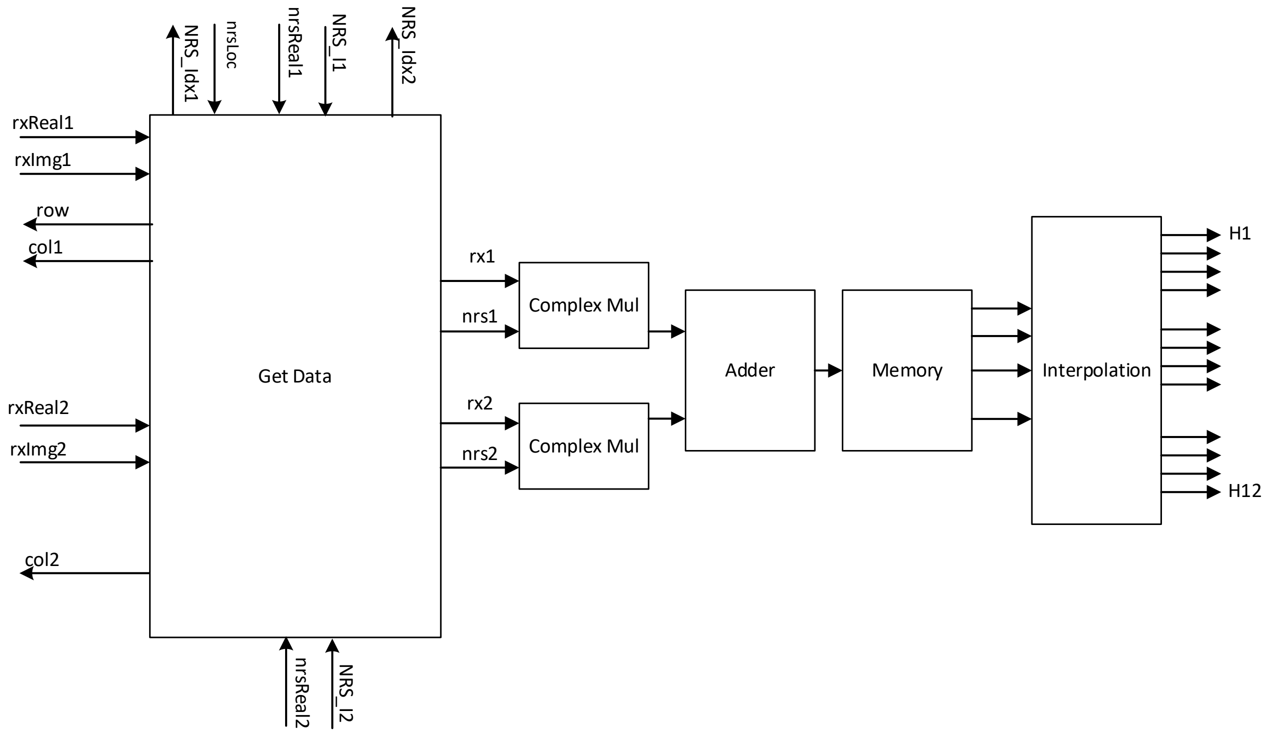


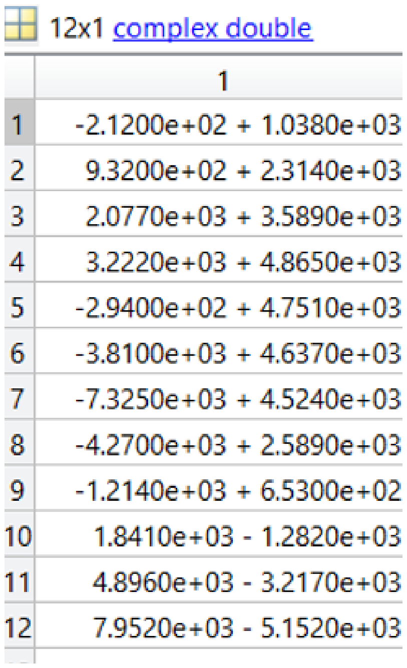
Figure [55]: Channel Estimation detailed hardware

The estimation process is done through the following steps:

1- We gather the needed data which is the received pilots (noisy) and the locally generated pilots (noiseless) the received are stored in the resource demapper and the generated in the NRS generation memory.
2- Calculate the channel frequency response by dividing which is implemented as multiplication by conjugate
3- Getting the average and store them in the memory
4- A combinational interpolation block to interpolate between the calculated four values to get the twelve-channel estimate.

3.6.3. Results

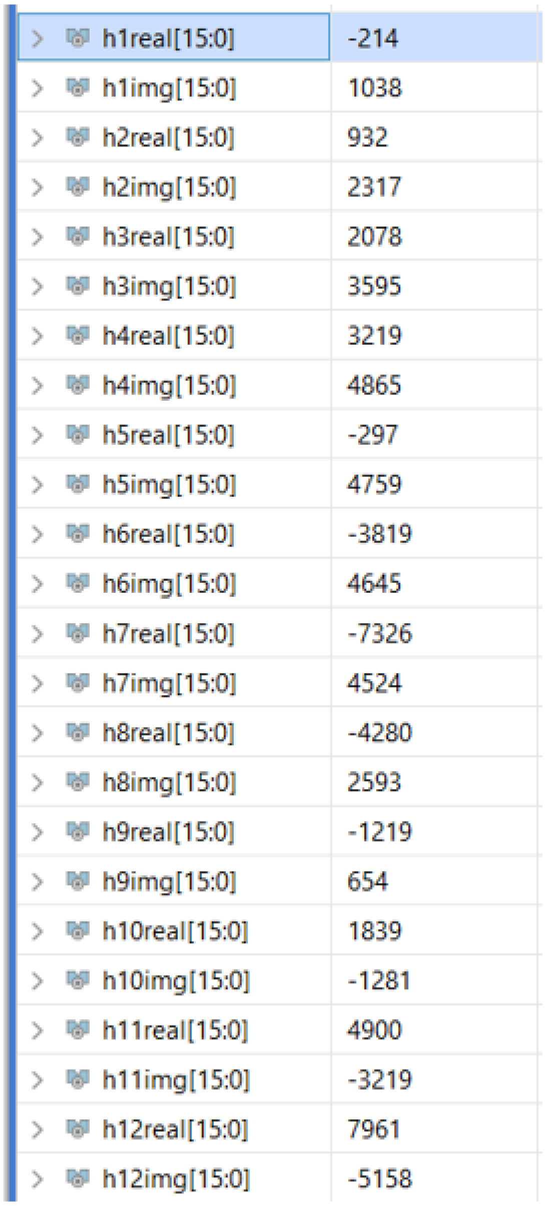
3.6.3.1. Matlab vs RTL



12x1 complex double

	1
1	-2.1200e+02 + 1.0380e+03
2	9.3200e+02 + 2.3140e+03
3	2.0770e+03 + 3.5890e+03
4	3.2220e+03 + 4.8650e+03
5	-2.9400e+02 + 4.7510e+03
6	-3.8100e+03 + 4.6370e+03
7	-7.3250e+03 + 4.5240e+03
8	-4.2700e+03 + 2.5890e+03
9	-1.2140e+03 + 6.5300e+02
10	1.8410e+03 - 1.2820e+03
11	4.8960e+03 - 3.2170e+03
12	7.9520e+03 - 5.1520e+03

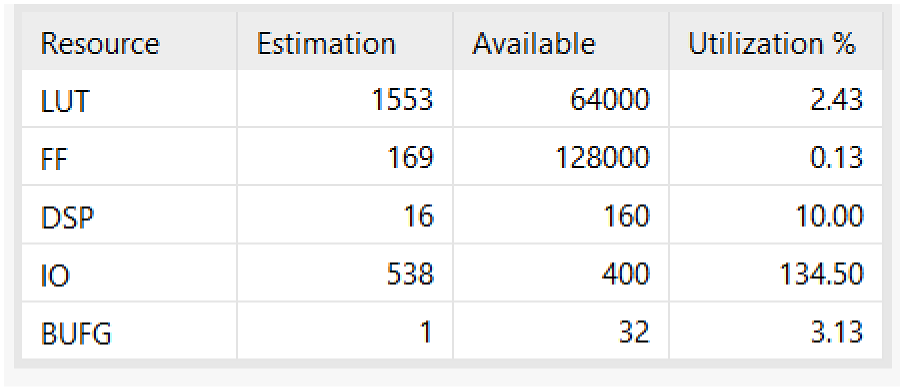
Figure [57]: Matlab estimates



h1real[15:0]	-214
h1img[15:0]	1038
h2real[15:0]	932
h2img[15:0]	2317
h3real[15:0]	2078
h3img[15:0]	3595
h4real[15:0]	3219
h4img[15:0]	4865
h5real[15:0]	-297
h5img[15:0]	4759
h6real[15:0]	-3819
h6img[15:0]	4645
h7real[15:0]	-7326
h7img[15:0]	4524
h8real[15:0]	-4280
h8img[15:0]	2593
h9real[15:0]	-1219
h9img[15:0]	654
h10real[15:0]	1839
h10img[15:0]	-1281
h11real[15:0]	4900
h11img[15:0]	-3219
h12real[15:0]	7961
h12img[15:0]	-5158

Figure [56]: RTL estimates

3.6.3.2. Synthesis Results



Resource	Estimation	Available	Utilization %
LUT	1553	64000	2.43
FF	169	128000	0.13
DSP	16	160	10.00
IO	538	400	134.50
BUFG	1	32	3.13

Figure [58]: Channel Estimation Synthesis results

3.7. NRS Value Generator

3.7.1. Top module

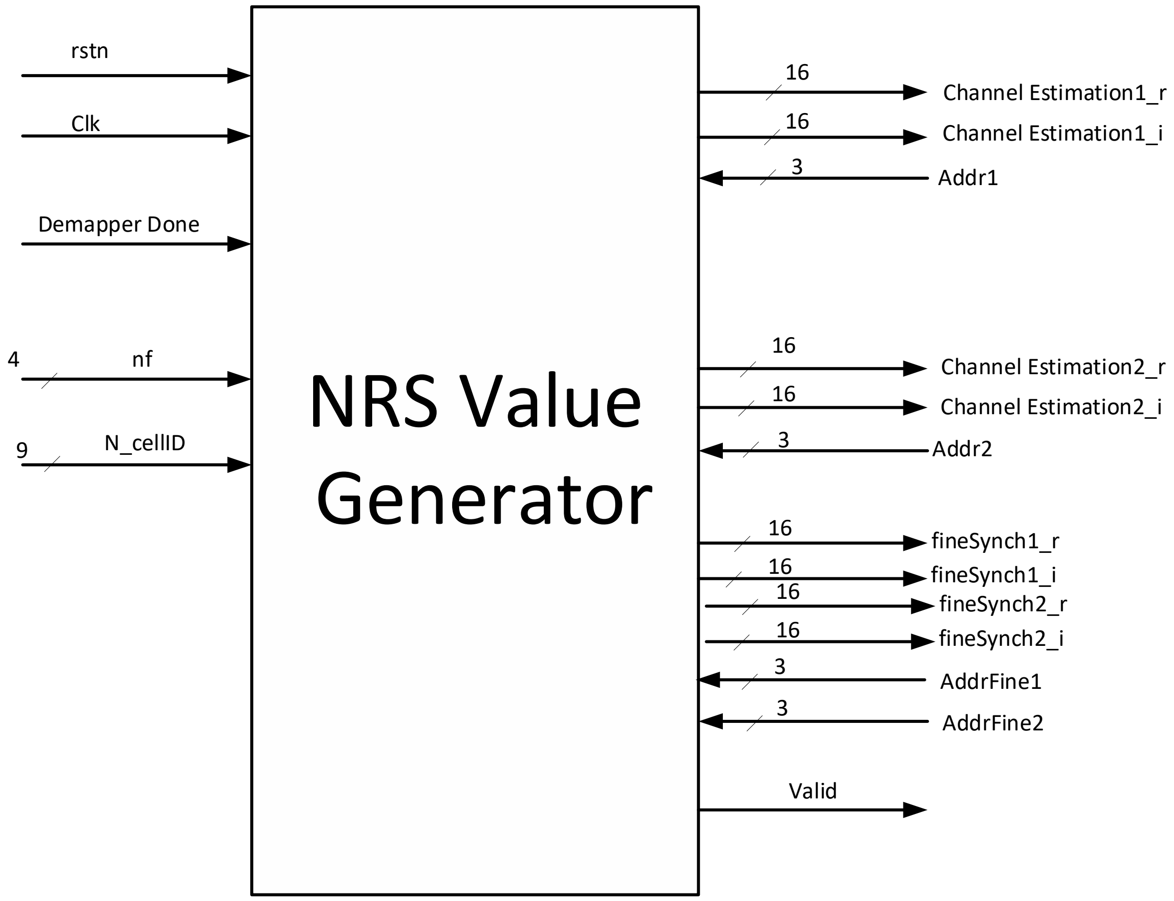


Figure [59]: NRS Value Generator top module

Table [16]: NRS Value Generator interface table

Signal Name	Direction	Width	Description
Clk	Input	1	130ns clock signal
Rstn	Input	1	Active low reset signal
demapperDone	Input	1	Signal indicates that storing subframe is done
Nf	Input	4	Subframe number
N_cellID	Input	9	Cell ID
Addr1	Input	3	Address of first pilot from channel estimation
Addr2	Input	3	Address of second pilot channel estimation
Addrfine1	Input	3	Address of first pilot from fine synchronization
Addrfine2	Input	3	Address of second pilot from fine synchronization
ChannelEstimation1_r	Output	16	First pilot real part
ChannelEstimation1_i	Output	16	First pilot imaginary part
ChannelEstimation2_r	Output	16	Second pilot real part
ChannelEstimation2_r	Output	16	Second pilot imaginary part

fineSynch1_r	Output	16	First pilot real part
fineSynch1_i	Output	16	First pilot imaginary part
fineSynch2_r	Output	16	Second pilot real part
fineSynch2_i	Output	16	Second pilot imaginary part
Valid	Output	1	Signal verifies that generation is done

3.7.2. Detailed Hardware

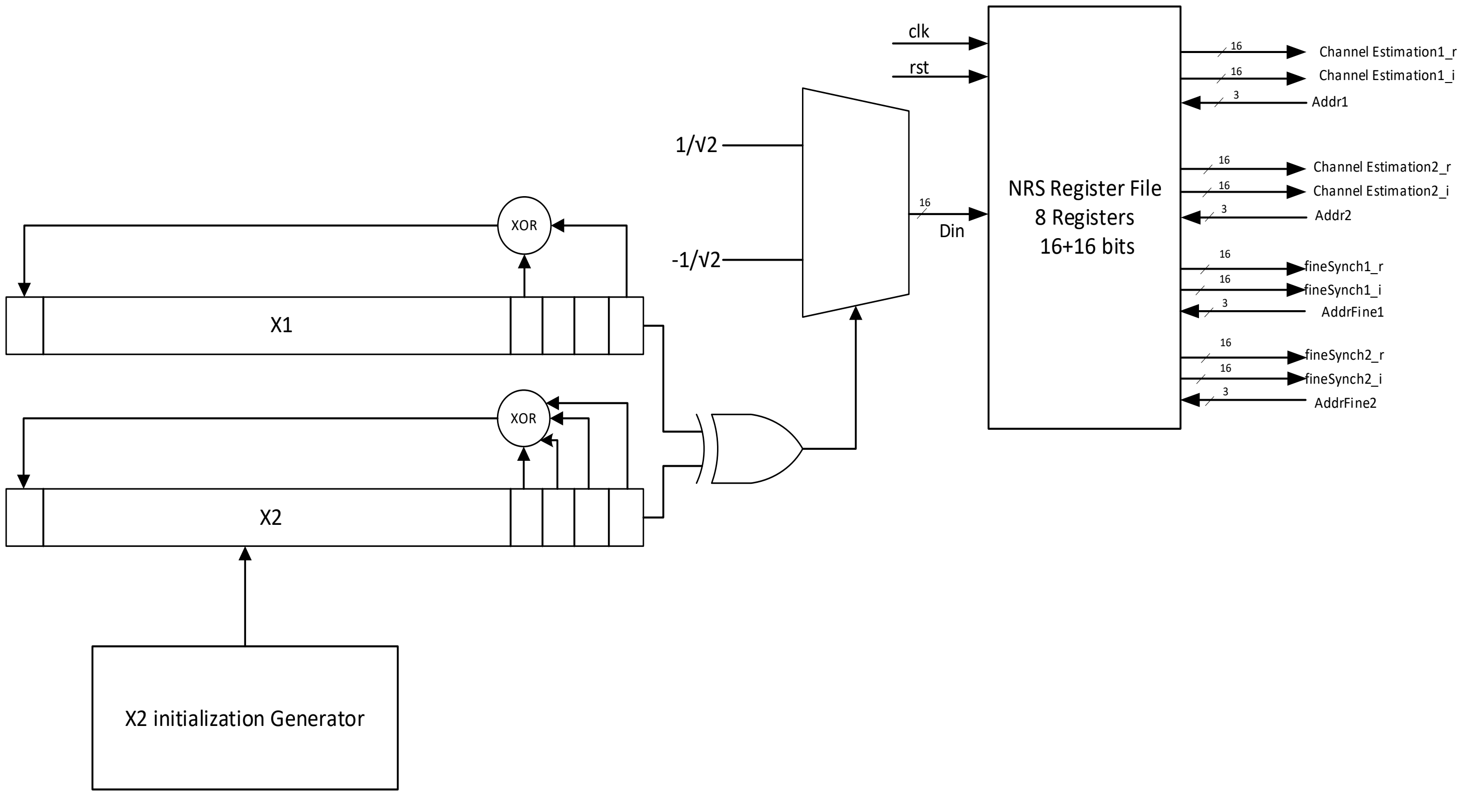


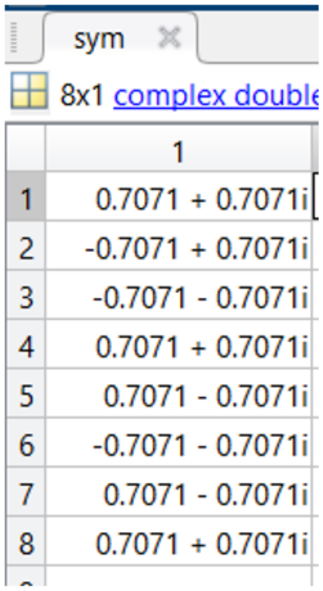
Figure [60]: NRS Value Generator detailed hardware

The pilot value is either $\frac{1}{\sqrt{2}}\, or\ -\frac{1}{\sqrt{2}}$ depending on the value of the golden sequence. So, x2 initialization generator generates the initializing sequence for $x2$ then both are XORed giving the golden sequence to choose which value is to be stored in the register file. Other blocks will interface with this register file accessing it with the required address to extract the needed generated pilot.

3.7.3. Results

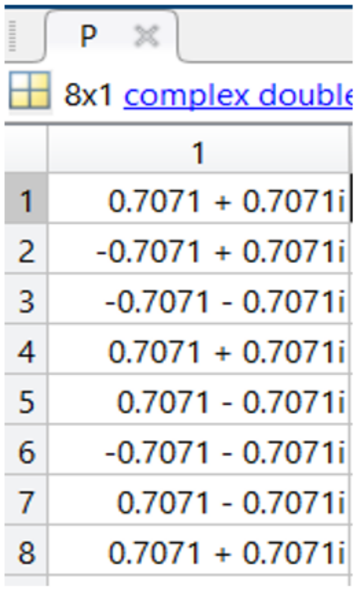
3.7.3.1. Matlab vs RTL

For subframe number = 0 and N_cellID = 1



sym
8x1 complex double

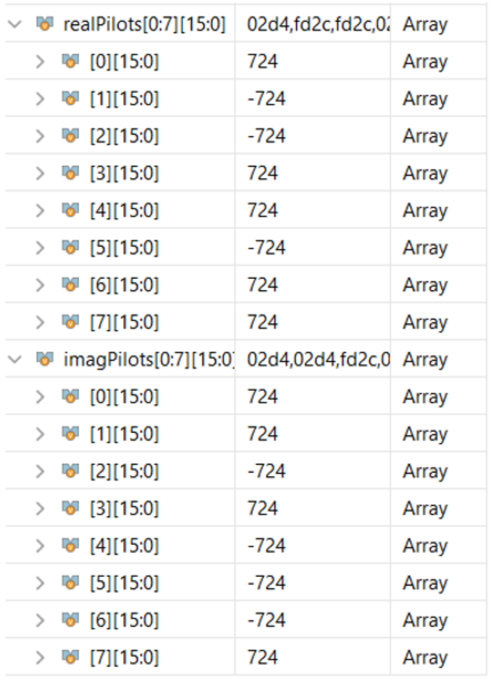
	1
1	0.7071 + 0.7071i
2	-0.7071 + 0.7071i
3	-0.7071 - 0.7071i
4	0.7071 + 0.7071i
5	0.7071 - 0.7071i
6	-0.7071 - 0.7071i
7	0.7071 - 0.7071i
8	0.7071 + 0.7071i



P
8x1 complex double

	1
1	0.7071 + 0.7071i
2	-0.7071 + 0.7071i
3	-0.7071 - 0.7071i
4	0.7071 + 0.7071i
5	0.7071 - 0.7071i
6	-0.7071 - 0.7071i
7	0.7071 - 0.7071i
8	0.7071 + 0.7071i

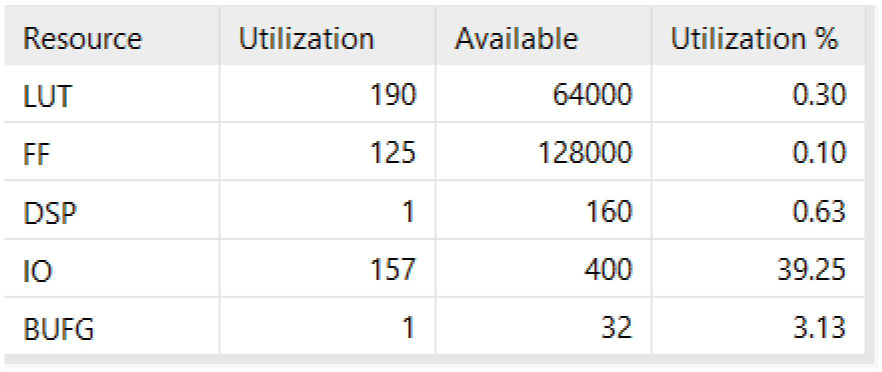
Figure [61]: MATLAB function vs Our function



realPilots[0:7][15:0]	02d4,fd2c,fd2c,02	Array
[0][15:0]	724	Array
[1][15:0]	-724	Array
[2][15:0]	-724	Array
[3][15:0]	724	Array
[4][15:0]	724	Array
[5][15:0]	-724	Array
[6][15:0]	724	Array
[7][15:0]	724	Array
imagPilots[0:7][15:0]	02d4,02d4,fd2c,0	Array
[0][15:0]	724	Array
[1][15:0]	724	Array
[2][15:0]	-724	Array
[3][15:0]	724	Array
[4][15:0]	-724	Array
[5][15:0]	-724	Array
[6][15:0]	-724	Array
[7][15:0]	724	Array

Figure [62]: RTL results

3.7.3.2. Synthesis Results



Resource	Utilization	Available	Utilization %
LUT	190	64000	0.30
FF	125	128000	0.10
DSP	1	160	0.63
IO	157	400	39.25
BUFG	1	32	3.13

Figure [63]: NRS Values Generator Synthesis results

3.8. NRS Index Generator

3.8.1. Top module

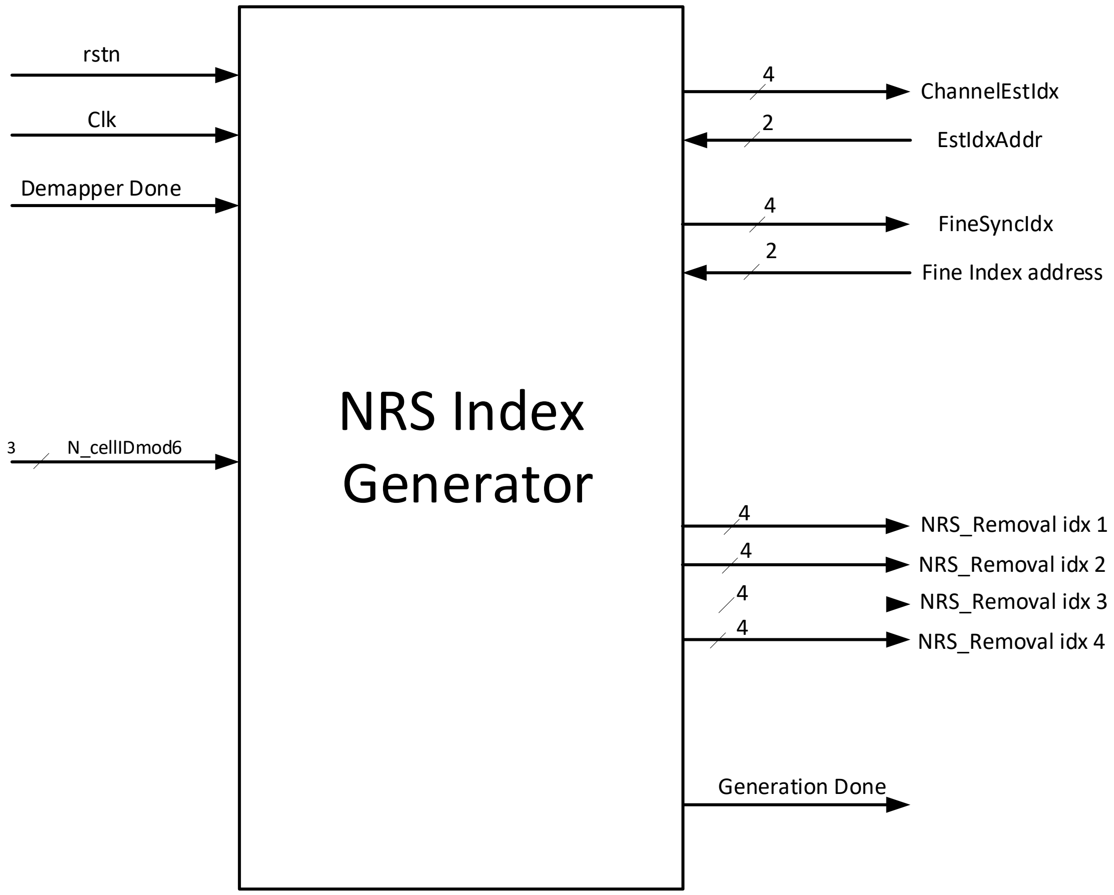


Figure [64]: NRS Index Generator top module

Table [17]: NRS Index Generator interface table

Signal Name	Direction	Width	Description
Clk	Input	1	520ns clock signal
Rstn	Input	1	Active low reset signal
demapperDone	Input	1	Signal indicates that storing subframe is done
N_cellIDmod6	Input	3	Cell ID modulus 6
EstIdxAddr	Input	2	Address of pilot needed by channel estimation
FineSyncAddr	Input	2	Address of pilot needed by fine synchronization
ChannelEstIdx	Output	4	Index of the pilot needed by channel est. in RB
FineSyncIdx	Output	4	Index of the pilot needed by fine sync. in RB
NRSremovalIdx1	Output	4	Index of first pilot to be given to NRS removal
NRSremovalIdx2	Output	4	Index of second pilot to be given to NRS removal
NRSremovalIdx3	Output	4	Index of third pilot to be given to NRS removal
NRSremovalIdx4	Output	4	Index of fourth pilot to be given to NRS removal
GenerationDone	Output	1	Signal verifies that index generation is done

3.8.2. Detailed hardware

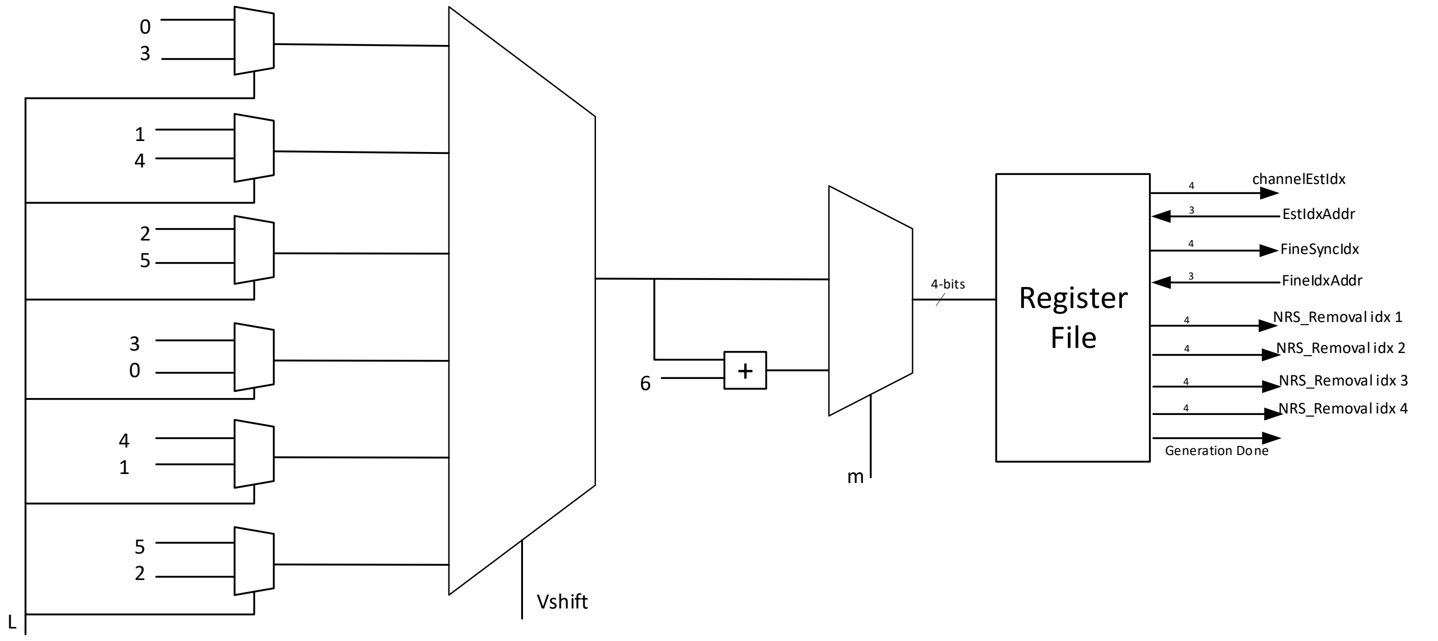


Figure [65]: NRS Index Generator detailed hardware

The design intends to implement the equations in the standard to calculate the value of the indices to extract them from the subframe

3.8.3. Design Challenges and Solutions

3.8.3.1. Calculations

Instead of using adders or any arithmetic block the design is implemented as multiplexer this is because the pilots are always in OFDM symbol number 5 or 6 and the rest of the equation are all set of choices from certain values

By that we reduce the consumed power using multiplexers instead of arithmetic units.

3.8.3.2. Memory

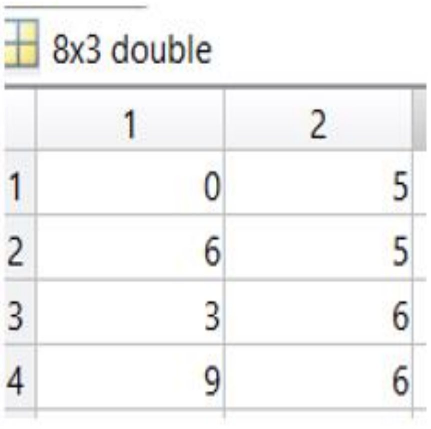
As the pilots are always in pair per subcarrier and always at OFDM symbol 5 or 6. So, we only need to know four values of the four subcarriers and there is no need to store eight values as every subcarrier of the calculated four has two pilots also no need to store the number of OFDM symbol as it is well known to any block in the system that the pilots are at OFDM symbol 5 and 6.

By that we have used half the size of the memory storing four numbers other than eight.

3.8.4. Results

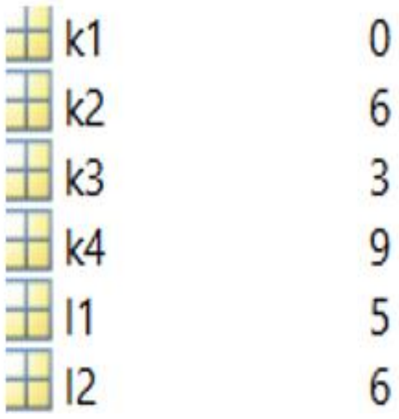
3.8.4.1. Matlab vs RTL

For subframe number = 1 and N_cellID = 0



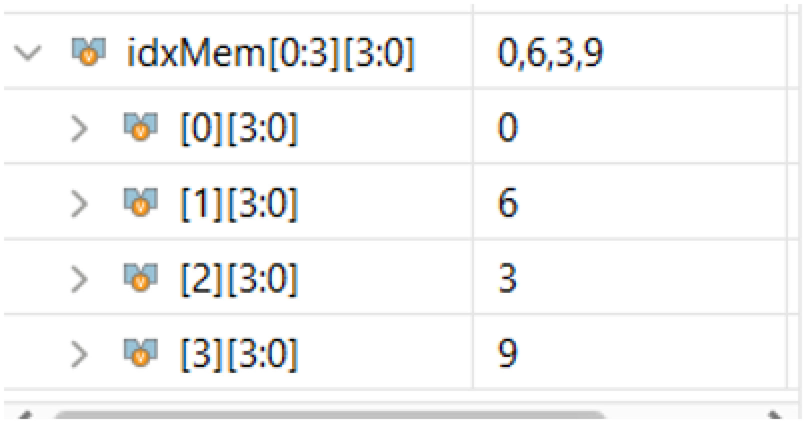
8x3 double

	1	2
1	0	5
2	6	5
3	3	6
4	9	6



k1	0
k2	6
k3	3
k4	9
l1	5
l2	6

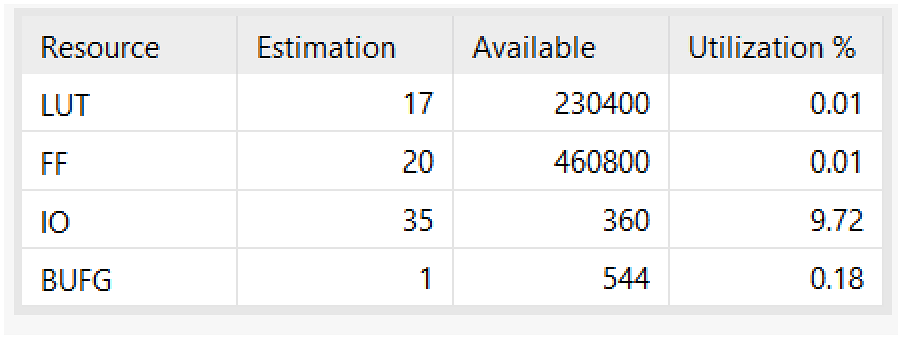
Figure [66]: MATLAB function vs Our function



idxMem[0:3][3:0]	0,6,3,9
[0][3:0]	0
[1][3:0]	6
[2][3:0]	3
[3][3:0]	9

Figure [67]: RTL results

3.8.4.2. Synthesis Results



Resource	Estimation	Available	Utilization %
LUT	17	230400	0.01
FF	20	460800	0.01
IO	35	360	9.72
BUFG	1	544	0.18

Figure [68]: NRS Index Generator Synthesis results

3.9. Channel Equalizer

3.9.1. Top module

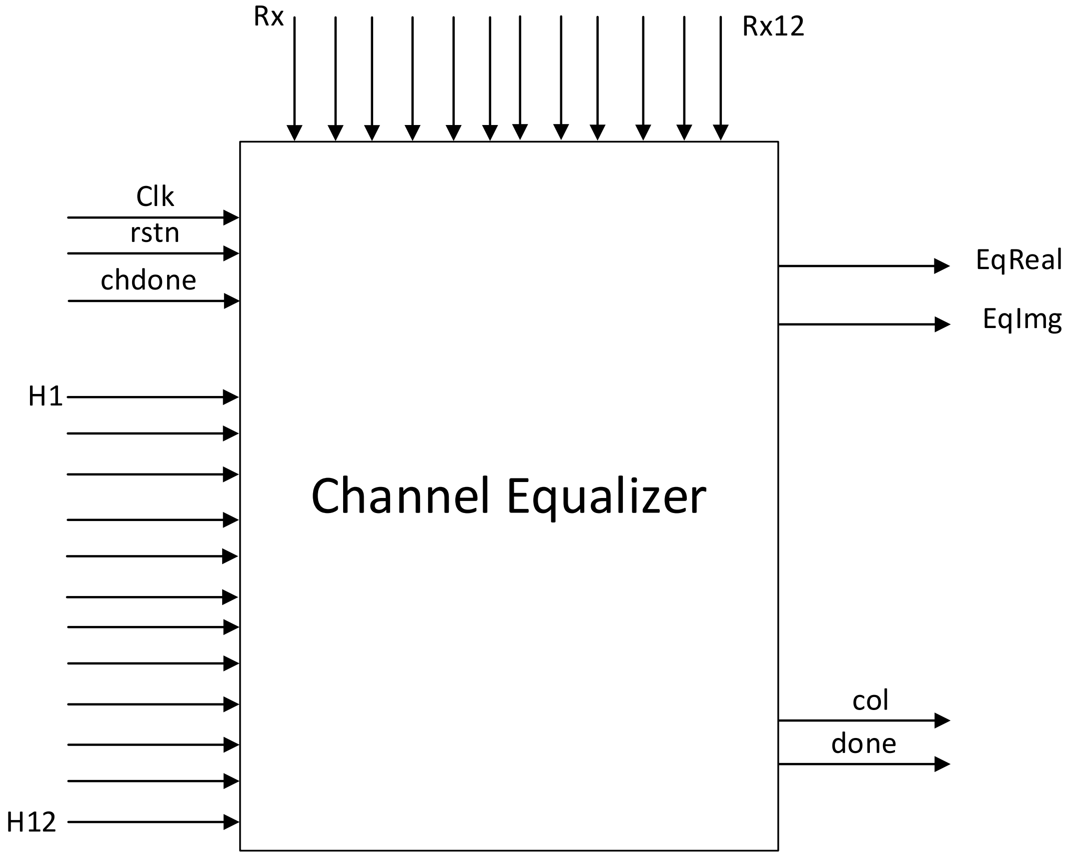


Figure [69]: Channel Equalizer top module

Table [18]: Channel Equalizer interface table

Signal Name	Direction	Width	Description
Clk	Input	1	520ns clock signal
Rstn	Input	1	Active low reset signal
chDone	Input	1	Signal indicates that channel estimation is done
H1 … H12	Input	16	Channel estimates per subcarrier
Rx1 … Rx12	Input	16	OFDM symbol to be equalized
Col	Output	4	Number of required OFDM symbol to be equalized
EqReal	Output	12	MSB of real part of the equalized symbol
EqImg	Output	12	MSB of imaginary part of the equalized symbol
Done	Output	1	Signal indicates that symbol has equalized

3.9.2. Detailed Hardware

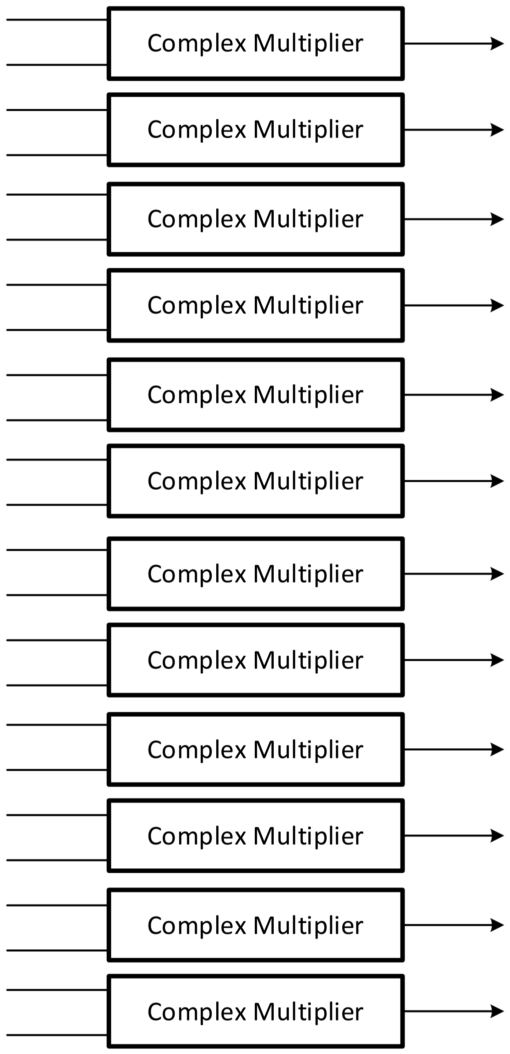


Figure [70]: Channel Equalizer detailed hardware

The design is a set of twelve complex multiplier (conjugate) to equalize the data symbols by diving them by the channel estimated value.

There is a counter that counts the number of equalized OFDM symbols, the value of the counter is sent to the resource demapper to extract the need OFDM symbol to equalized.

Also, this counter value will be used to detect the last column as we need to know when we reached the last OFDM symbol to achieve the timing needed in Parallel to Serial block.

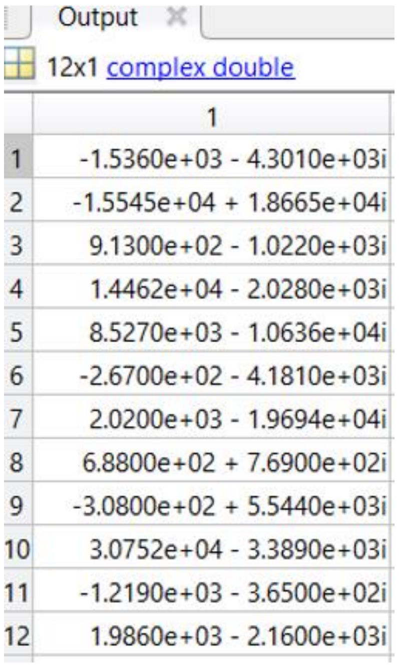
3.9.3. Design Challenges and Solutions

There was a challenge in communication between this block and parallel to serial block because the data are stored in the resource demapper for certain time which doesn't include the overhead of converting each symbol to serial and this conversion takes around 12 clock cycles.

The counter is passed to the parallel to serial and it gets registered there to be used as the storing time of RB is enough for all Symbols except the last one.

3.9.4. Results

3.9.4.1. Matlab vs RTL

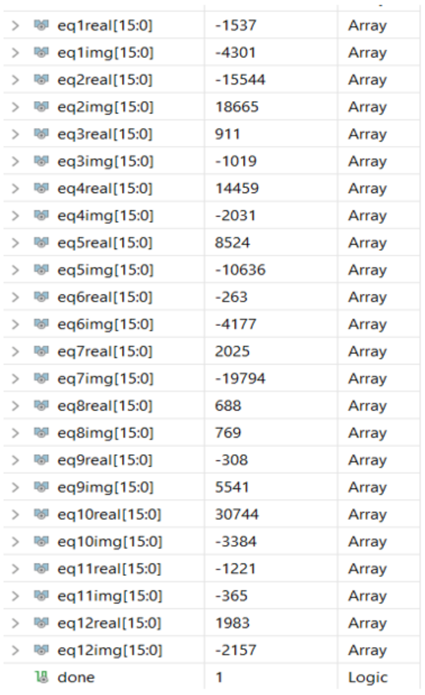


Output

12x1 complex double

	1
1	-1.5360e+03 - 4.3010e+03i
2	-1.5545e+04 + 1.8665e+04i
3	9.1300e+02 - 1.0220e+03i
4	1.4462e+04 - 2.0280e+03i
5	8.5270e+03 - 1.0636e+04i
6	-2.6700e+02 - 4.1810e+03i
7	2.0200e+03 - 1.9694e+04i
8	6.8800e+02 + 7.6900e+02i
9	-3.0800e+02 + 5.5440e+03i
10	3.0752e+04 - 3.3890e+03i
11	-1.2190e+03 - 3.6500e+02i
12	1.9860e+03 - 2.1600e+03i

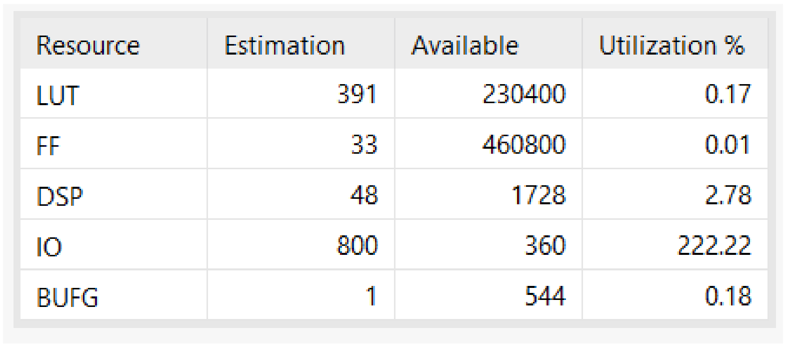
Figure [72]: Matlab equalized OFDM symbol



eq1real[15:0]	-1537	Array
eq1img[15:0]	-4301	Array
eq2real[15:0]	-15544	Array
eq2img[15:0]	18665	Array
eq3real[15:0]	911	Array
eq3img[15:0]	-1019	Array
eq4real[15:0]	14459	Array
eq4img[15:0]	-2031	Array
eq5real[15:0]	8524	Array
eq5img[15:0]	-10636	Array
eq6real[15:0]	-263	Array
eq6img[15:0]	-4177	Array
eq7real[15:0]	2025	Array
eq7img[15:0]	-19794	Array
eq8real[15:0]	688	Array
eq8img[15:0]	769	Array
eq9real[15:0]	-308	Array
eq9img[15:0]	5541	Array
eq10real[15:0]	30744	Array
eq10img[15:0]	-3384	Array
eq11real[15:0]	-1221	Array
eq11img[15:0]	-365	Array
eq12real[15:0]	1983	Array
eq12img[15:0]	-2157	Array
done	1	Logic

Figure [71]: RTL equalized OFDM symbol

3.9.4.2. Synthesis results



Resource	Estimation	Available	Utilization %
LUT	391	230400	0.17
FF	33	460800	0.01
DSP	48	1728	2.78
IO	800	360	222.22
BUFG	1	544	0.18

Figure [73]: Equalizer Synthesis results

3.10. Parallel to Serial and NRS removal

3.10.1. Top module

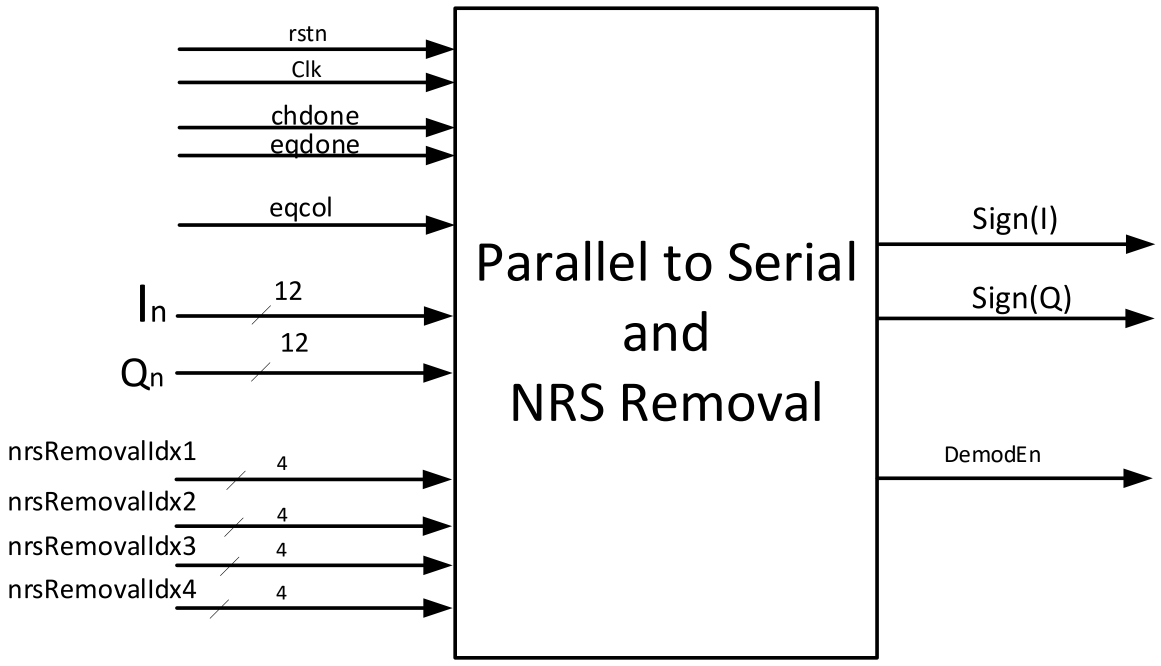


Figure [74]: Parallel to Serial and NRS removal top module

Table [19]: Parallel to Serial and NRS removal interface table

Signal Name	Direction	Width	Description
Clk	Input	1	520ns clock signal
Rstn	Input	1	Active low reset signal
chDone	Input	1	Signal indicates that channel estimation is done
Eqdone	Input	1	Signal indicates that channel equalization is done
In	Input	12	MSB of the real part of the equalized symbol (sign)
Qn	Input	12	MSB of the imaginary part of the equalized symbol (sign)
Eqcol	Input	4	Number of equalized OFDM symbol
nrsRemovalIdx1	Input	4	Index of first pilot to be removed
nrsRemovalIdx2	Input	4	Index of second pilot to be removed
nrsRemovalIdx3	Input	4	Index of third pilot to be removed
nrsRemovalIdx4	Input	4	Index of fourth pilot to be removed
Signi	Output	1	Sign of real part
SignQ	Output	1	Sign of imaginary part
DemodEn	Output	1	Enable signal to bit processing indicate valid data

As the demodulation is hard demodulation so it needs only the sign of the QPSK symbol the equalizer send us 12 bits representing the sign of every QPSK symbol in the OFDM symbol. Here we take these bits in payload shift register and start outputting this bits in serial.

3.10.2. Design Challenges and Solutions

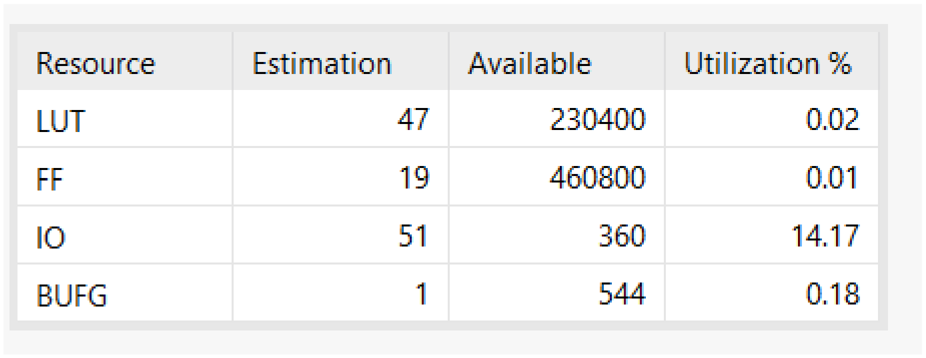
The main challenge was in handling the last OFDM symbol as the time of storing the subframe doesn't include converting it to serial data.

The time was sufficient to convert every OFDM symbol except for the last one for that the design has special registers to store the needed values for the last column.

We register the last set of QPSK symbols to be output in serial then manage to disable the enable after all the subframe has been converted.

3.10.3. Results

3.10.3.1. Synthesis results



Resource	Estimation	Available	Utilization %
LUT	47	230400	0.02
FF	19	460800	0.01
IO	51	360	14.17
BUFG	1	544	0.18

Figure [75]: Parallel to Serial and NRS removal Synthesis results

3.11. Fine Synchronizer

3.11.1. Top Module

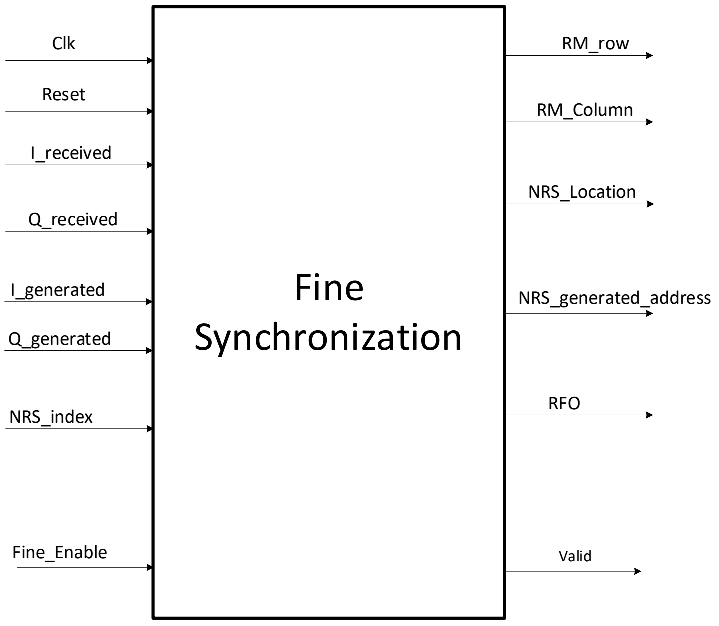


Figure [76]: Fine Synchronization Top Module

Table [20]: Fine Synchronization Interface Table

Signal Name	Direction	Width	Description
Clk_260	Input	1	3.84 MHz clock signal
Reset	Input	1	Global reset signal
Fine_Enable	Input	1	Enable signal for the Fine Synchronization module from the NRS Value Generator
I_received1	Input	16	Real value of the first-time domain received NRS signal
Q_ received1	Input	16	Imaginary value of the first-time domain received NRS signal
I_received2	Input	16	Real value of the second-time domain received NRS signal
Q_ received2	Input	16	Imaginary value of the second-time domain received NRS signal

I_generated1	Input	16	Real value of the first-time domain generated NRS signal
Q_ generated1	Input	16	Imaginary value of the first-time domain generated NRS signal
I_generated2	Input	16	Real value of the second-time domain generated NRS signal
Q_ generated2	Input	16	Imaginary value of the second-time domain generated NRS signal
NRS_index	Input	4	NRS index from the NRS Index Generator block
RM_row1	Output	4	Row address to the Resource De-Mapper to get the first received NRS
RM_Column1	Output	4	Column address to the Resource De-Mapper to get the first received NRS
RM_row2	Output	4	Row address to the Resource De-Mapper to get the second received NRS
RM_Column2	Output	4	Column address to the Resource De-Mapper to get the second received NRS
NRS_Location	Output	3	NRS Location to the NRS Index Generator block
NRS_generated_address1	Output	3	NRS address to the NRS Value Generator block to get the second received NRS
NRS_generated_address2	Output	3	NRS address to the NRS Value Generator block to get the second received NRS
RFO	Output	19	Residual Frequency offset to the CFO Block
valid	Output	1	Valid signal that verifies the integrity of the output bus data

3.11.2. Detailed Hardware

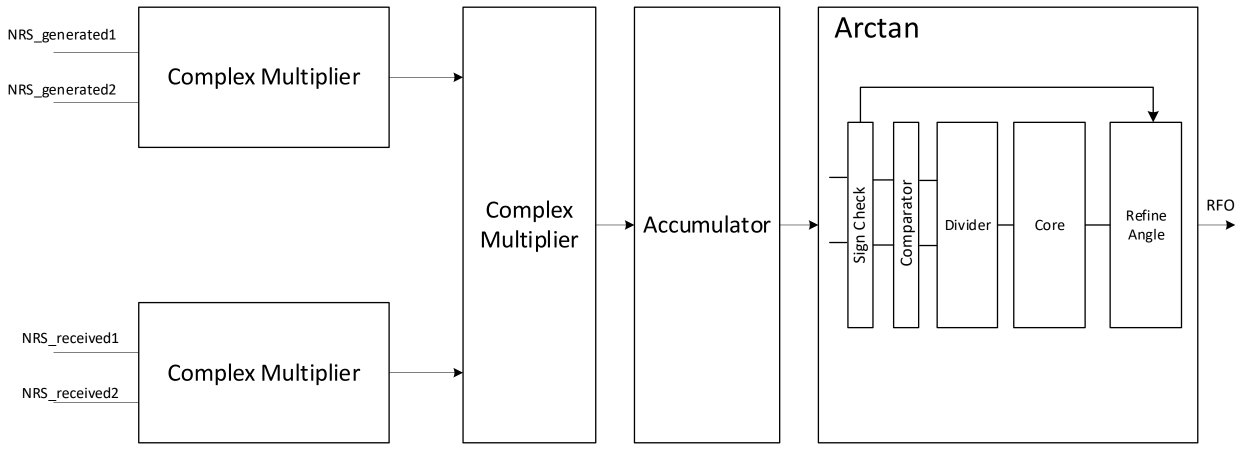


Figure [77]: Fine Synchronization Detailed Design

Fine Synchronization Description:

- Fetching each pair takes place as
  1. The block first sends the grid Location of this pair to the NRS address generation block.
  2. The NRS address generation block sends back the row indices of these pairs.
  3. The block then sends the NRS address in the Resource Element De-Mapper as row and column addresses in order to fetch the received NRS. And also sends the NRS address to the locally generated NRS memory in the NRS value generation block.
- Each pair of received NRS is complex multiplied as well as each pair of locally generated NRS signals
- The received complex product result is divided by the generated complex product result, but instead of division we use complex multiplication by using the complex conjugate. There is no need to account for the magnitude in this operation because the complex division will not affect the magnitude as it results in unity factor.
- The above operations are repeated for four iterations and each time the accumulator accumulates the resulting phase
- Frequency offset can be extract by using the Arctan function

Arctan Synchronization Description:

The Low power implementation of the arctan is using the linear range of the arctan function in order to obtain the real arctan value.

z= [0,1]

θ= [0:45]

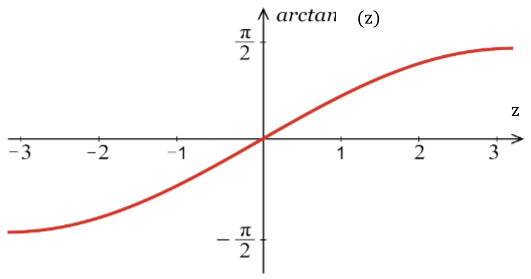


Figure [78]: Arctan Linear Range

The implementation of the arctan is divided into 5 stages

- Checking the sign of the real and imaginary parts of the complex quantity that we are calculating the arctan of and storing their sign to be used later in the Redefine angle stage.
- Passing the absolute values of the x (real part) and y (imaginary part), compare them and the smaller one is then used as numerator for the division operation while the greater is the numerator. This is in order to guarantee that the division z quantity is in range of [0,1].
- As the numerator and denominator of the division are identified in the comparator stage, this stage divides them using Non-restoring Algorithm
- Core is the main stage of this arctan which uses the value of this division to get the result of the angle from [0:45] using the following equations:

$$\tan^{-1} z = \begin{cases} 56z, & 0 < z < 0.25 \\ 50z+1.5, & 0.25 < z < 0.5 \\ 40z + 6.5, & 0.5 < z < 0.75 \\ 32z + 13, & 0.75 < z < 1 \end{cases}$$

- If the numerator was the real part, x<y then $\theta = 90 - \tan^{-1} z$
- If the numerator was the imaginary part, y<x then $\theta = \tan^{-1} z$
- The resulted θ lies in the first quadrature in range of [0:90]

- In the Refine angle block, final value of θ_{final} is determined through the full range of [0:360] using the stored signs in the first stage using the following equations:

$$\theta_{final} = \begin{cases} \theta, & x+ve,\ y+ve \\ 180-\theta, & x-ve,\ y+ve \\ \theta-180, & x-ve,\ y-ve \\ -\theta, & x+ve,\ y-ve \end{cases}$$

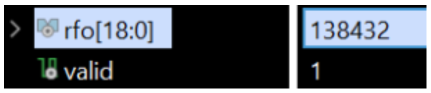
The Calculated RFO value is a 19 bits result used by the CFO Correction block. 10 for the fraction and 9 for the integer.

3.11.3. Design Challenges and Solutions

The non-restoring algorithm divides absolute numbers and gives the final results based on this. So, it will not work well with the used fixed-point representation. The solution was to shift right the denominator by 10 as the fixed-point representation has 10 bits for the fraction and 6 bits for the integer. That's how it is guaranteed that the quotient which is the final division result is following the fixed-point representation.

3.11.4. Results

3.11.3.1. MATLAB vs. RTL



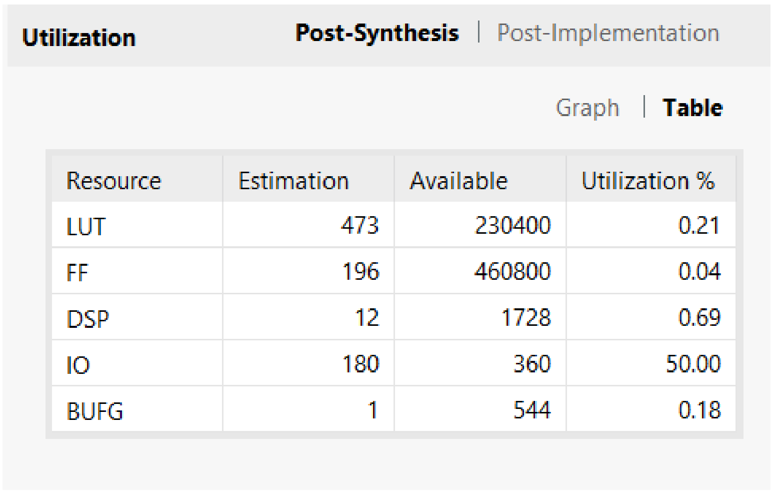
```
rfo[18:0]   138432
valid       1
```

```
RFO_calculated =

  135.5205


RFO_matlab =

  135.4698
```

Figure [79]: Fine Synchronizer RTL Results

138432 in fixed point representation. To convert it, 138432/1024 = 135.1875.

3.11.3.2. Synthesis Results



Utilization **Post-Synthesis** | Post-Implementation

Graph | **Table**

Resource	Estimation	Available	Utilization %
LUT	473	230400	0.21
FF	196	460800	0.04
DSP	12	1728	0.69
IO	180	360	50.00
BUFG	1	544	0.18

Figure [80]: Fine Synchronizer Synthesis Results

3.12. Demodulator

3.12.1. Top Module

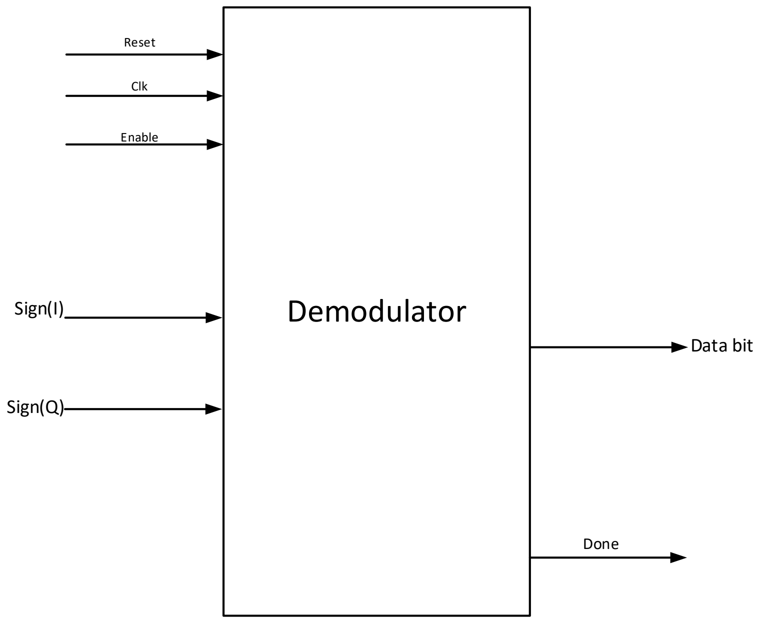


Figure [81]: Demodulator Top Module

Table [21]: Demodulator Interface Table

Signal Name	Direction	Width	Description
Clk_260	Input	1	3.84 MHz clock signal
Reset	Input	1	Global reset signal
Enable	Input	1	Enable signal for the Demodulator module to start operating
Sign_I	Input	1	sign of In-phase component received from the P/S block
Sign_Q	Input	1	sign of Quadrature component received from the P/S block
Data_bit	Output	1	Serial data bits output to descrambler
Done	Output	1	signal that verifies the integrity of the output bus data

3.12.2. Detailed Hardware

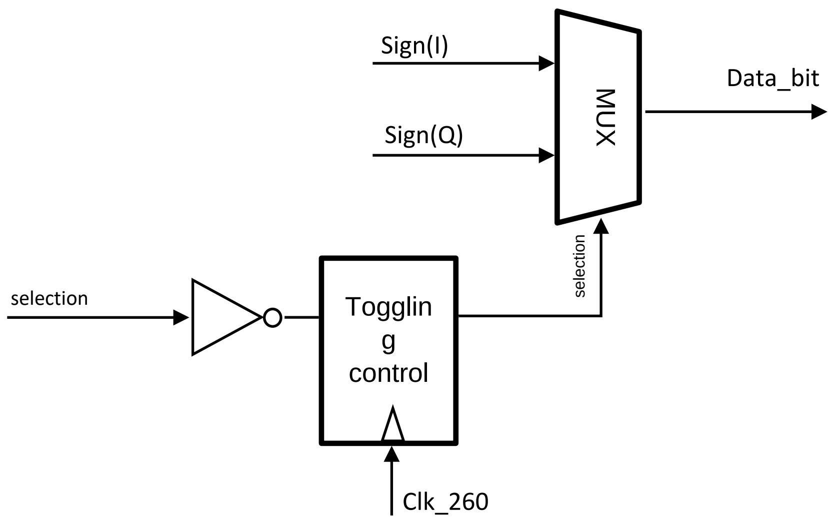


Figure [82]: Demodulator Detailed Hardware

3.12.3. Results

All the upcoming MATLAB results are for a transport block size of 24 bits that consists of 12 bits of zeros and 12 bits of ones.

3.12.3.1. MATLAB vs. RTL

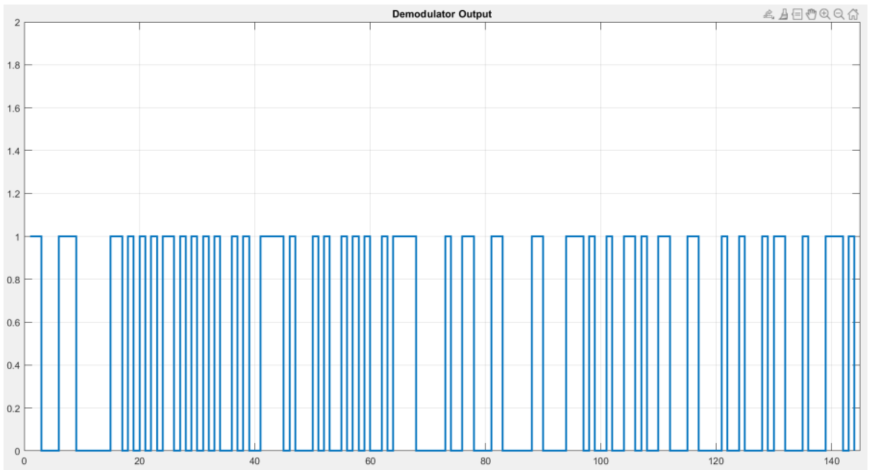


Figure [83]: Demodulator MATLAB Results

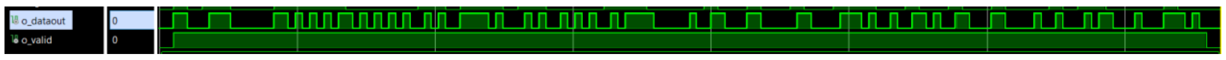
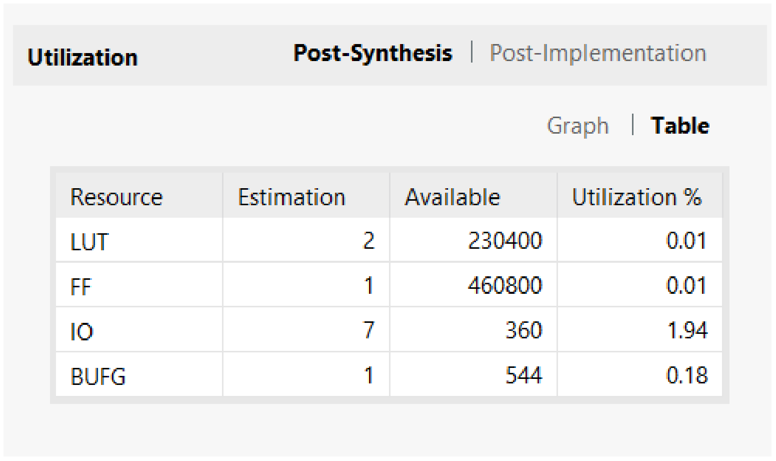


Figure [84]: Demodulator RTL Results

3.12.3.2. Synthesis Results



Resource	Estimation	Available	Utilization %
LUT	2	230400	0.01
FF	1	460800	0.01
IO	7	360	1.94
BUFG	1	544	0.18

Figure [85]: Demodulator Synthesis Results

3.13. Descrambler

3.13.1. Top Module

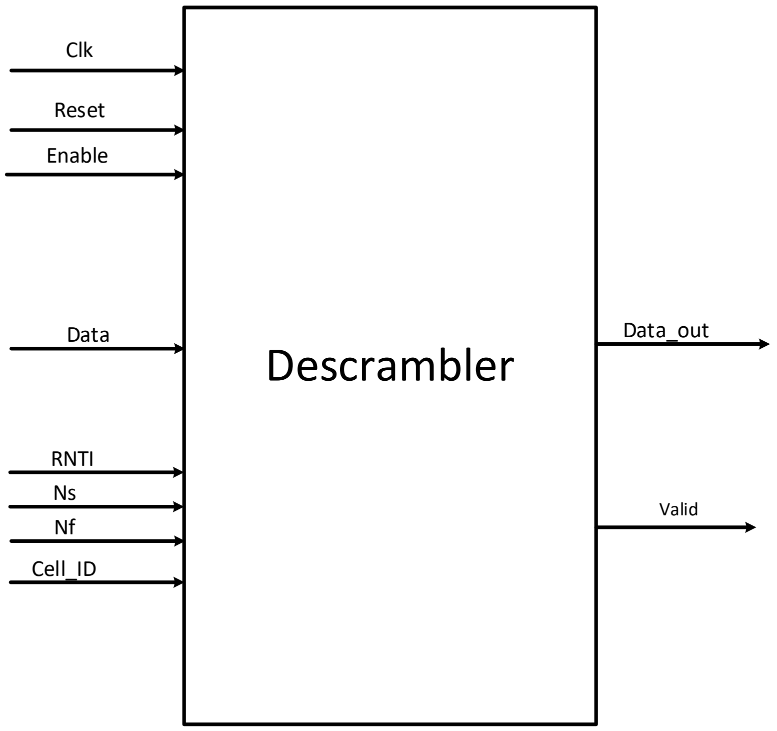


Figure [86]: Descrambler Top Module

Table [22]: Descrambler Interface Table

Signal Name	Direction	Width	Description
Clk_260	Input	1	3.84 MHz clock signal
Reset	Input	1	Global reset signal
Enable	Input	1	Enable signal for the Scrambler module to start operating
Data	Input	1	Serial input data bits from demodulator
RNTI	Input	16	Radio Network Temporary Identifier which is an upper layer parameter
Ns	Input	5	First slot of transmission
Nf	Input	1	First frame of transmission

Nsf	Input	4	Number of subframes used in codeword
Nrep	Input	12	Number of repetitions used in codeword
Cell_ID	Input	9	The Cell Identifier which is an upper layer parameter
Data_out	Output	1	Serial output data bits to the rate matcher
Valid	Output	1	signal that verifies the integrity of the output bus data

3.13.2. Detailed Hardware

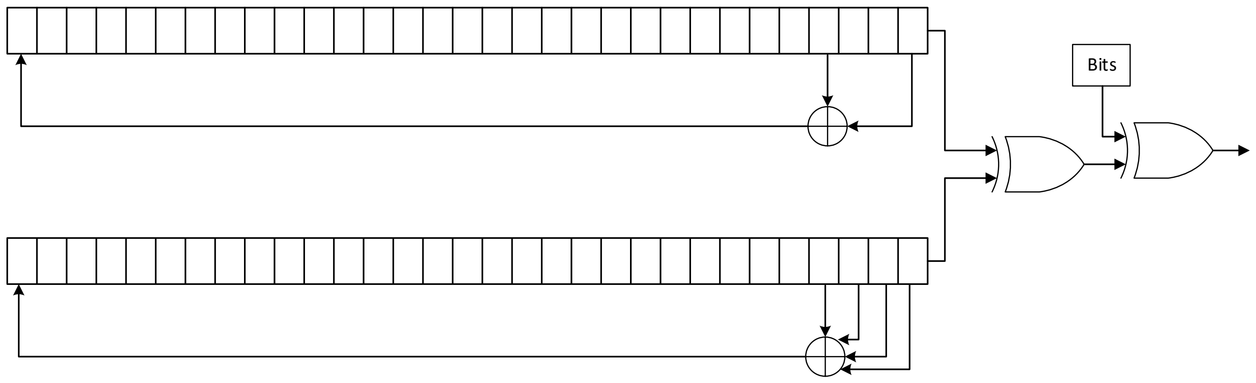


Figure [87]: Descrambler Detailed Design

Description: The two linear feedback shift registers are initialized with their values and then they take 1600 clock cycle to reach the desired golden sequence than should be XOR-ed with the input bitstream, until it finishes the received codeword from the demodulator. the codeword that is successive repetitions of the codeword subframes.

The linear feedback shift registers are re-initialized every $\min(N_{rep}, 4)$ transmissions of the codeword.

3.13.3. Results

TBS = 24, N_{ID}^{cell} = 1, n_f = 20, n_s= 0, n_{RNTI} = 1000

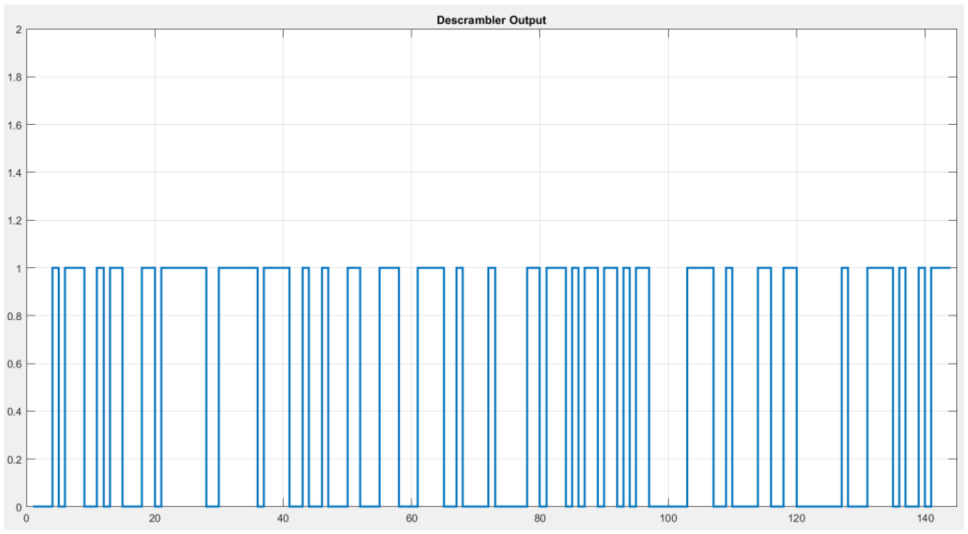


Figure [88]: Descrambler MATLAB Results

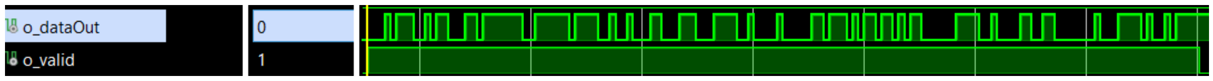
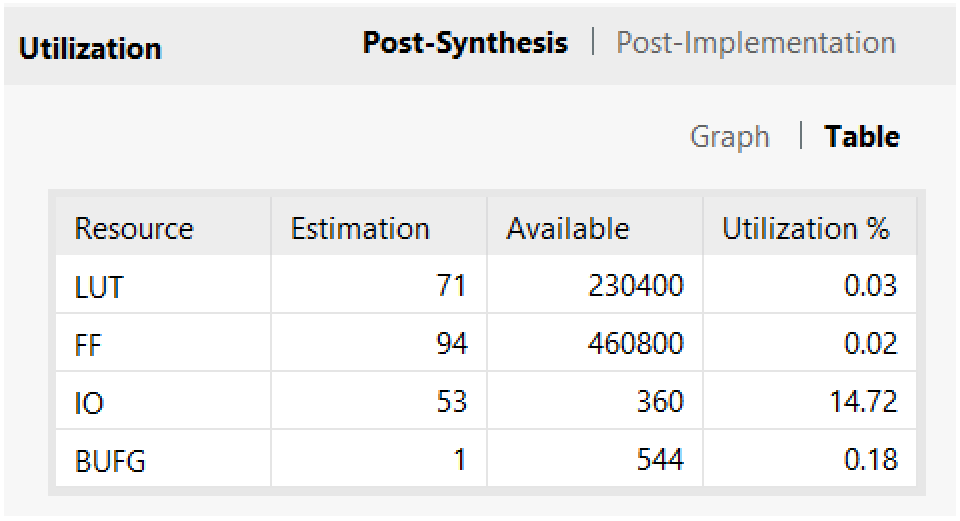


Figure [89]: Descrambler RTL Results



Utilization **Post-Synthesis** | Post-Implementation

Graph | **Table**

Resource	Estimation	Available	Utilization %
LUT	71	230400	0.03
FF	94	460800	0.02
IO	53	360	14.72
BUFG	1	544	0.18

Figure [90]: Descrambler Synthesis Results

3.14. Rate De-Matcher

3.14.1. Top Module

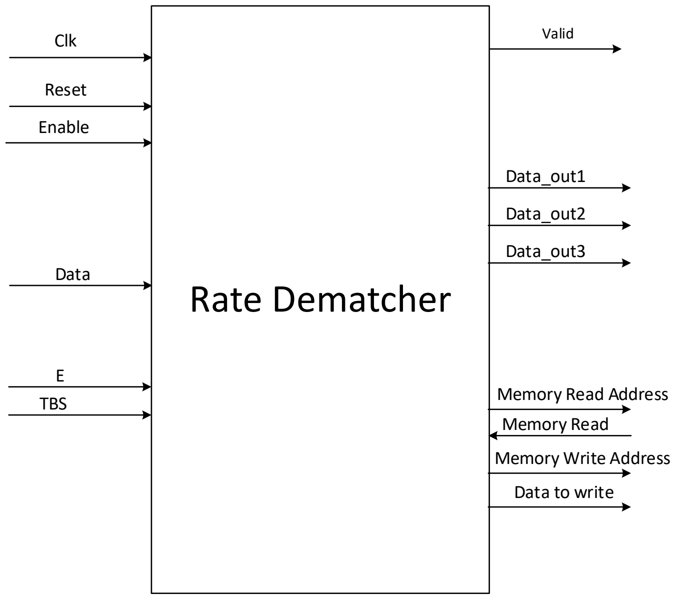


Figure [79]: Rate Dematcher Top Module

Table [23]: Rate De-Matcher Interface Table

Signal Name	Direction	Width	Description
Clk_260	Input	1	3.84 MHz clock signal
Clk_130	Input	1	7.69 MHz clock signal
Reset	Input	1	Global reset signal
Enable	Input	1	Enable signal for the Rate De-matcher.
Data	Input	1	Serial input data bits from Descrambler.
E	Input	24	Size of the input data to the Block.
TBS	Input	12	Transport Block Size of the incoming data
Matcher_repeat	Input	1	If decode didn’t decode correctly at first trial
Data_out1	Output	1	First output to decoder
Data_out2	Output	1	Second output to decoder
Data_out3	Output	1	Third output to decoder
Valid	Output	1	signal that verifies the integrity of the output bus data

3.14.2. Detailed Hardware

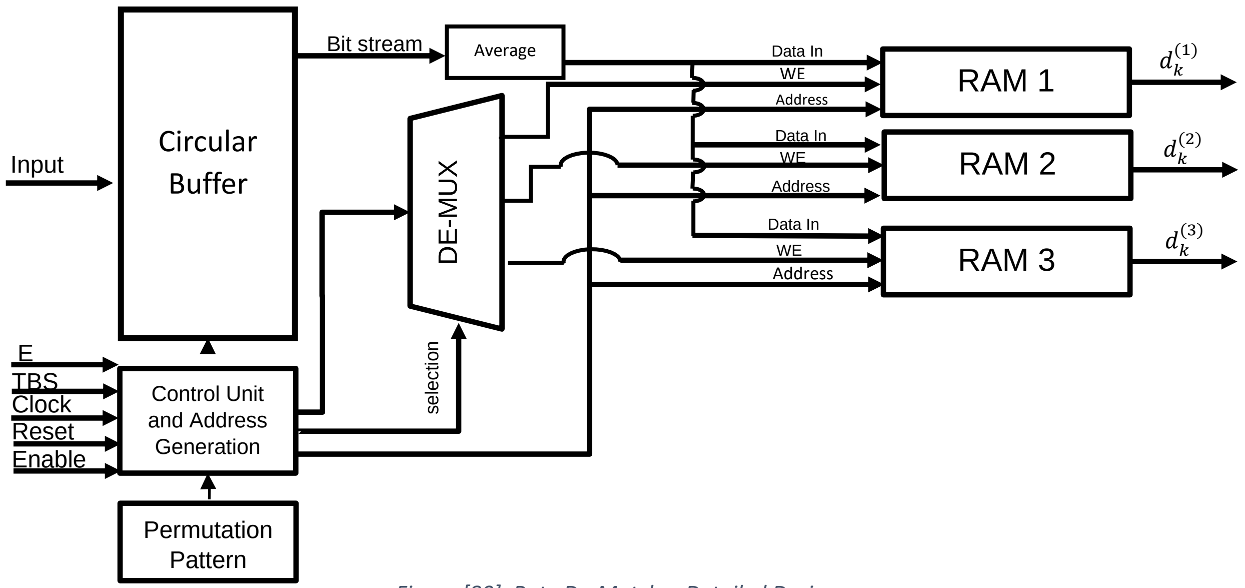


Figure [80]: Rate De-Matcher Detailed Design

Description:

- The input data is collected at the bit collecting memory whenever there's an enable signal high which indicates that the input from the descrambler is valid.
- Bits are written to the first memory until reach the circular buffer length. If the incoming stream is not over, the write pointer rolls over to the start of the circular buffer again to read the memory content and add the incoming input stream to them.
- when from the input data length E is received, bit collection memory output is averaged by dividing them by the number of repetitions and taking the majority vote.
- The data after majority vote is stored in each RAM each at a time, and the inter-column permutation matrix is performed by changing the address using the control unit.
- The control unit generated the addresses based on the ROM storing the intercolumn permutation pattern. And it also generates the write enable for each memory
- when the 3 RAMs are filled, they are read each row at a time without the dummy bits and the output is given to the decoder.

The Rate De-matcher control unit executes the following state machines in order to perform the block operations correctly without interfering with the reception of the next transport block as the reception is continuous in time. So, to handle this the control unit is built as two separate finite state machines that once the first once finishes it triggers the operation of the second state machine which perform the interleaving process and the output.

Address generation unit handles the three RAMs filling in order to account for the first N_D dummy bits and then store data at the right locations. Memory unrolling is performed in order to make the memory easier with only one address decoder instead of two row and column decoders.

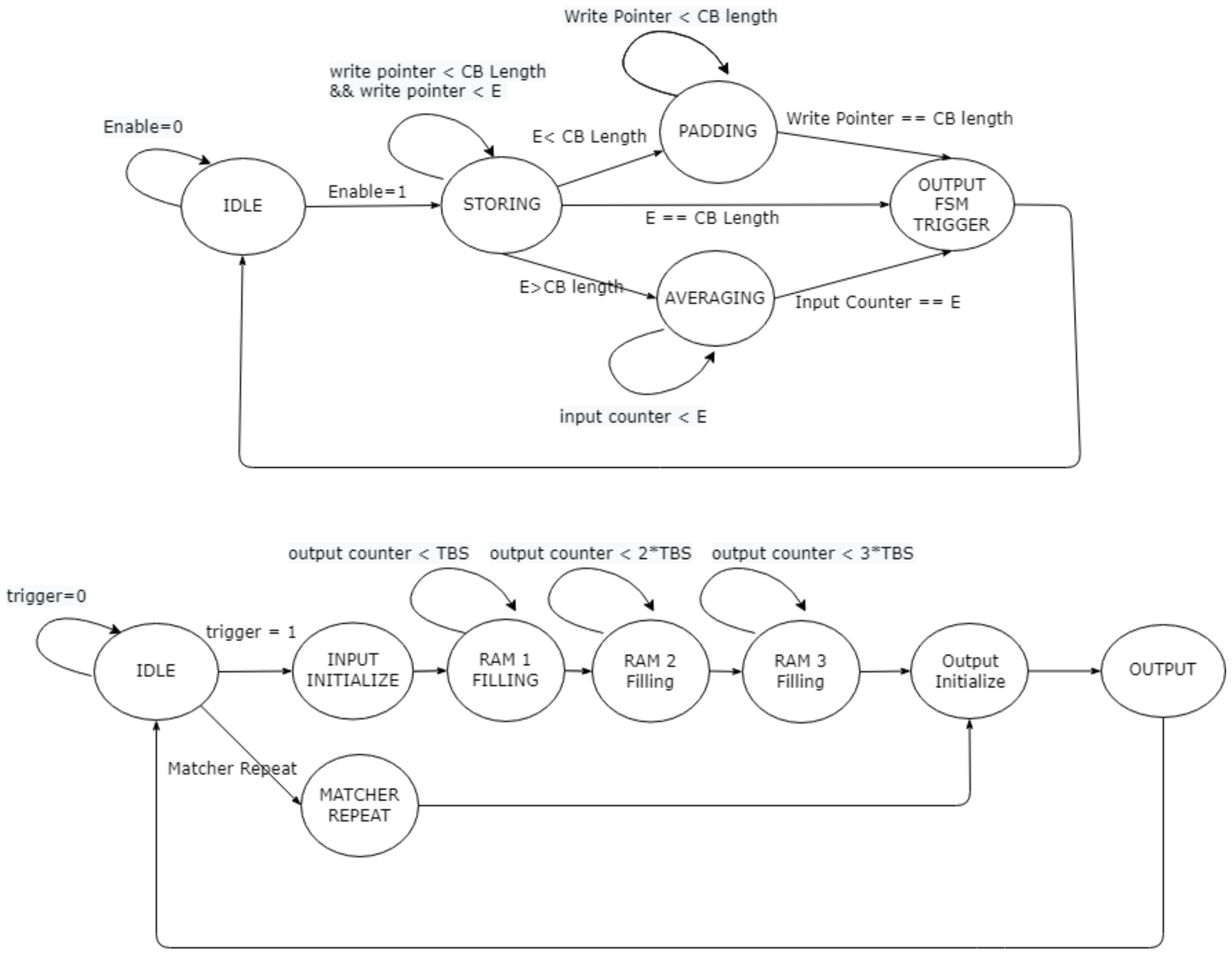


Figure [91]: Rate De-Matcher Control Finite State Machines

3.14.3. Design Challenges and Solutions

The block's first memory in the bit collecting stage cannot stop for the time interval that the three rams are being filled as the received subframes are continuous in time and every new subframe has data that propagates down the chain and needs to be stored correctly. In order to fix this the first memory has to be a dual port memory so the reading and writing processes are decoupled from each other. The first port is read/write port and the second port is a read only port.

A new problem arises from this implementation that in case of repetitions the memory has to read first the memory content and then add the incoming input to the stored value so for this to happen the first port has to operate at twice the frequency of the block with flock clock of 130 nsec. This solution was chosen because this new clock as already generated and used in our system chain at different other blocks so no overhead would appear from such scenario

3.14.4. Results

3.14.4.1. MATLAB v. RTL

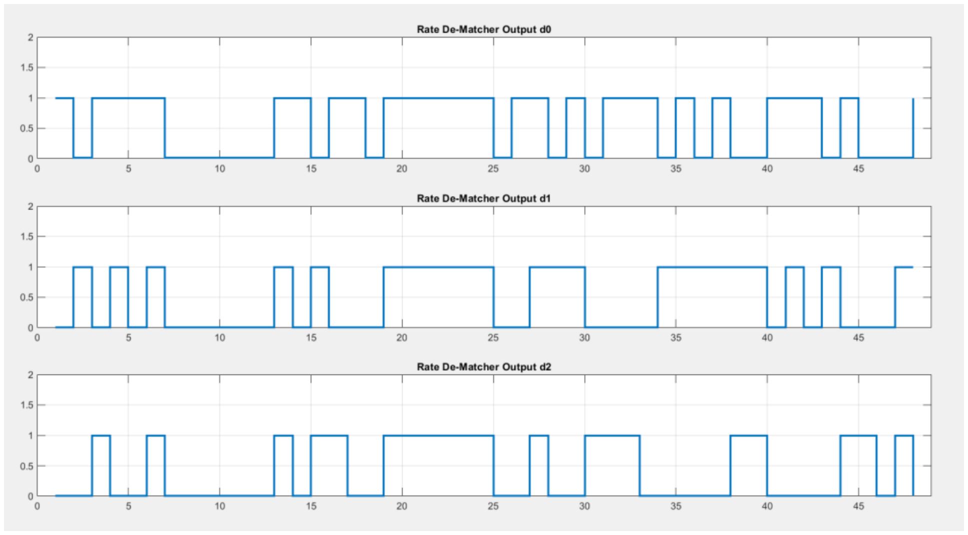


Figure [92]: Rate De-Matcher MATLAB Results

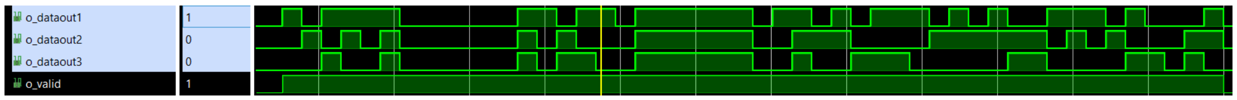
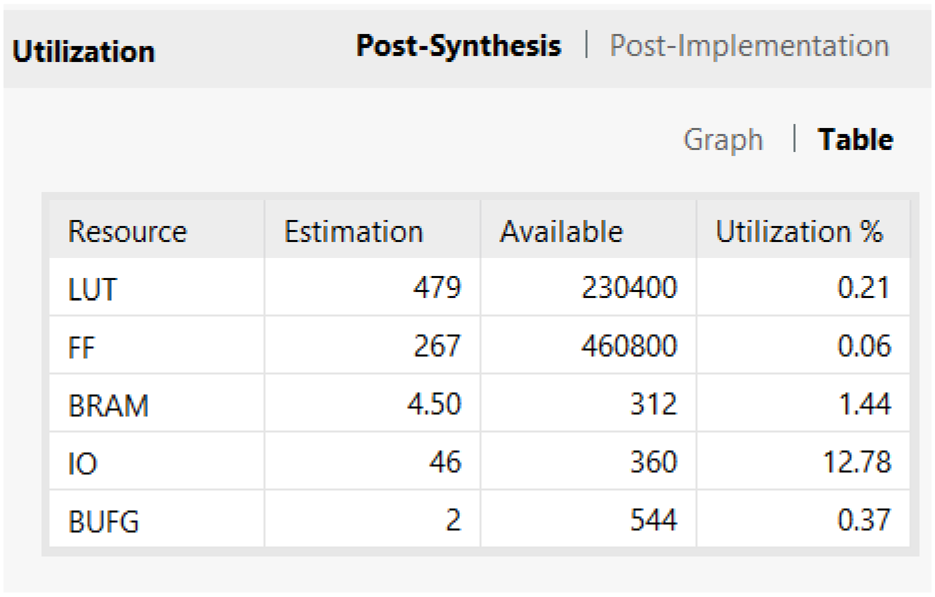


Figure [93]: Rate De-Matcher RTL Results

3.14.4.2. Synthesis Results



Resource	Estimation	Available	Utilization %
LUT	479	230400	0.21
FF	267	460800	0.06
BRAM	4.50	312	1.44
IO	46	360	12.78
BUFG	2	544	0.37

Figure [94]: Rate De-Matcher Synthesis Results

3.15. Viterbi Decoder

3.15.1. Block Interface

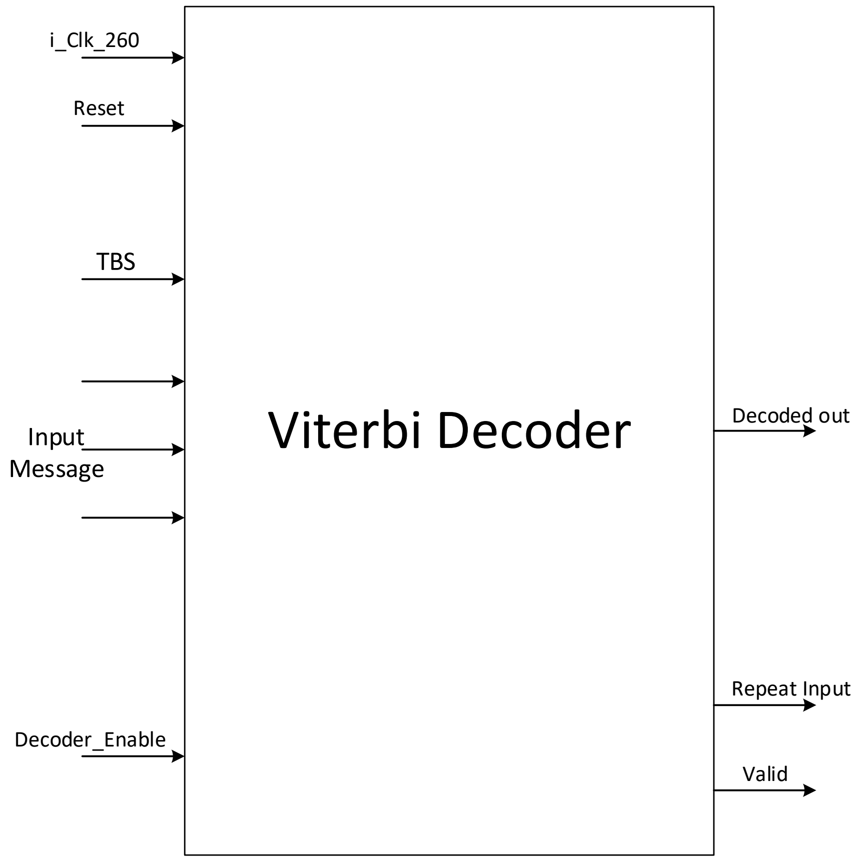


Figure [95]: Decoder Interface

Table [24]: Decoder Interface Table

Signal Name	Direction	Width	Description
Clk	Input	1	Clock signal to the block (260ns)
Reset	Input	1	Reset signal to the block
TBS	Input	12	Transport Block Size
Input Message	Input	3	Inputs divided to 3 parallel bits
Decoder_Enable	Input	1	Signal to enable the decoder
Decoded_out	Output	1	Output serial decoded bits
Valid	Output	1	Valid signal to next block
Repeat_Input	Output	1	Signal to previous block to repeat the input for another iteration

3.15.2. Detailed Hardware

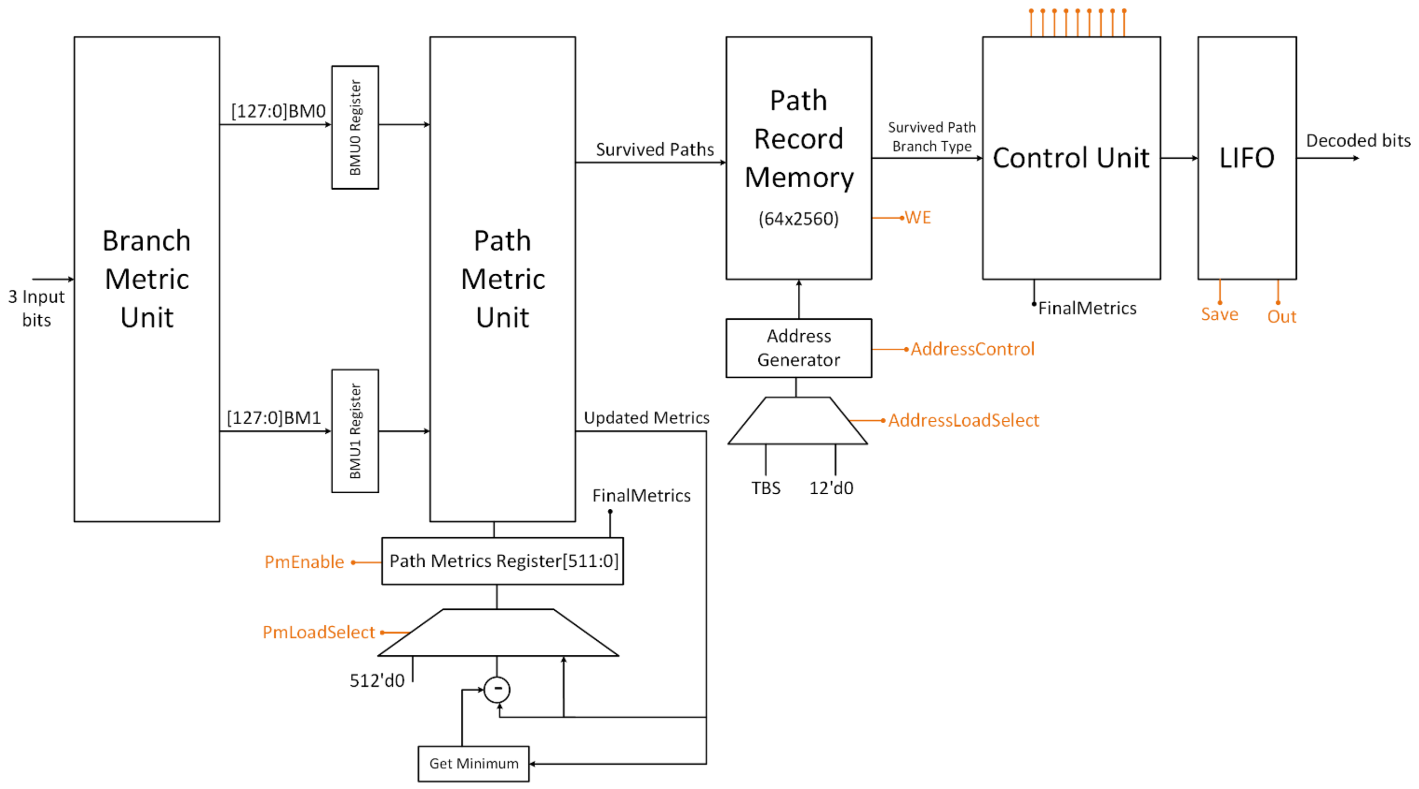


Figure [96]: Detailed hardware of the decoder

3.15.2.1. Branch Metric Unit (BMU)

This block is responsible for calculating the hamming distance between the input 3 bits and the expected outputs from all branches in the trellis diagram. Every state has two output branches as shown in the figure. This means that we need 128 Hamming Distance Units for 64 states.

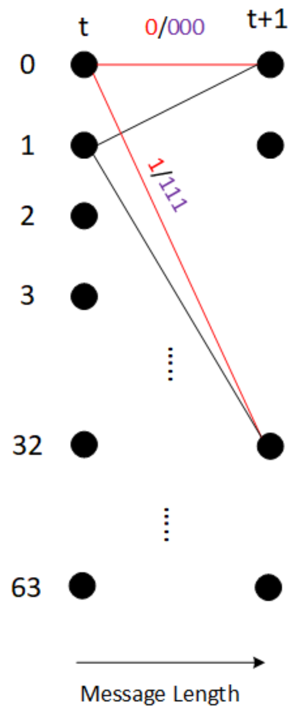


Figure [97]: Trellis Diagram

3.15.2.2. Path Metric Unit (PMU)

This block contains Add-Compare-Select units that work in this following flow:

1. Add the branch metrics to the saved path metrics.
2. Compares the two input paths to the next states.
3. Selects the path with higher metric.
4. Accumulates this metric in the path metrics memory.
5. Saves the selected path (high or low branch) in the path record memory.

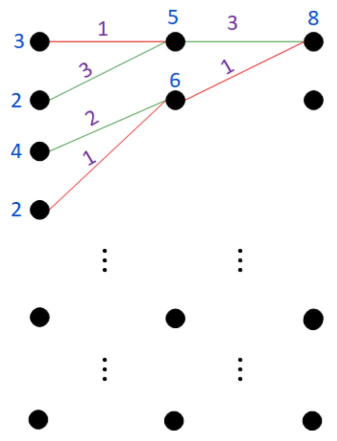


Figure [98]: Example of Path Metric Unit operation

In the shown figure, we can see an example of the path metric unit operation. The example shows that there's two different paths that are entering the same next state. The key to differentiate between them is to compare the path metrics of them after adding the new branch metrics to the path metrics then selecting the path that has the larger metric to be our survived path in this state. For this example, at the left side, we can see that the survived path that enters the first state is the path that has all green branches.

3.15.2.3. Path Record Memory

This block is a memory that saves the transitions in each state and used in the traceback unit. The size of the memory is 64x2560 bits as we need to store the condition of the branch entering the state whether it's the upper branch (0 is stored) or the lower branch (1 is stored). The encoding of the stored values will be elaborated in the next section.

3.15.2.4. Control Unit

It controls the entire operation of the decoding algorithm using a finite state machine that has these operations:

- Manages the calculation and saving of the survived paths in the path record memory.
- Performs the traceback operation by evaluating the winning path and reading from the path record memory.
- Checking the Tailbiting condition and controls the output flow from the LIFO memory.
- Performs the Wrap Around Viterbi Algorithm (WAVA) elaborated before.

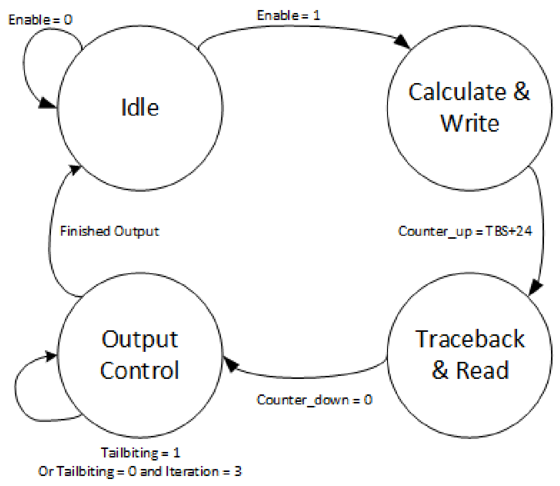


Figure [99]: Finite State Machine implemented in the control unit

3.15.2.5. LIFO Memory

A LIFO Memory of size (1x2560) is implemented to store the decoded bits from the traceback operation. Last-In-First-Out mechanism is required as the traceback operation starts from the last bit of the message to the start bit of the message.

3.15.3. Hardware Challenges and Solutions

3.15.3.1. Path Metrics are Monotonically Increasing

As the decoder proceeds in the trellis structure, the path metrics are being accumulated due to the Add-Compare-Select (ACS) operation. Moreover, the values are monotonically increasing as the branch metrics are positive values from 0 to 3. Also, given the fixed size of the path metrics registers (8 bits), these registers will overflow and the decoder will not operate correctly.

To avoid the overflow of these registers, in the shown figure, a simple hardware that is controlled by the control unit is added to subtract the minimum value stored in these registers occasionally before the overflow occurs. This avoids the overflow problem and also will not affect the operation as it only cares about the relation between the path metrics not the actual values of them.

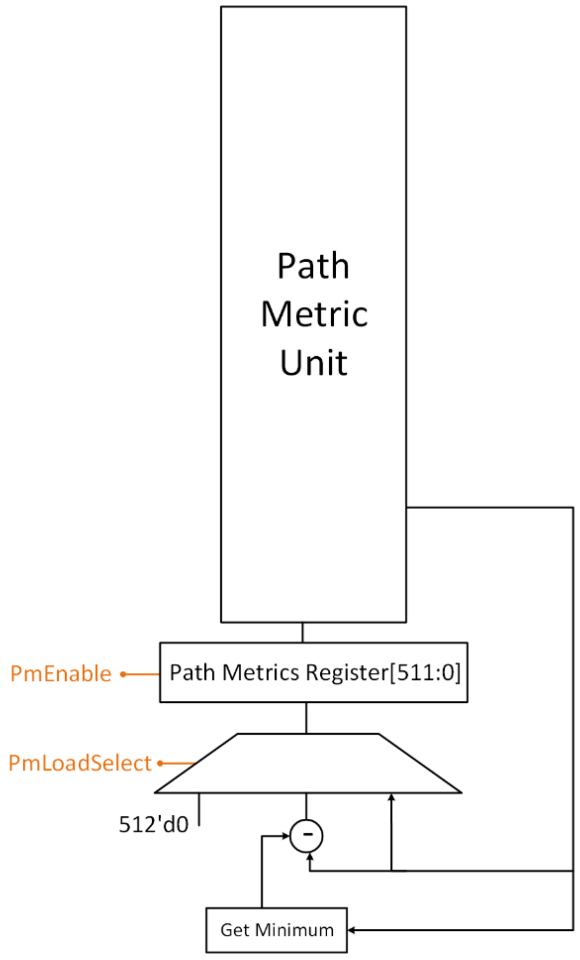


Figure [100]: Hardware added to avoid overflow

3.15.3.2. Path Record Memory Size

Traceback operation is done by knowing the branch and the two states connected to it in order to find the actual decoded bit and to go to the previous state. These parameters can be calculated in the first stages but they need to be stored as the traceback operation must start from the end of the message to the start of it. We can calculate the size of the memory using:

$$Memory\ Size = n_{states} * log_2 n_{states} * message\ length$$

In our case, the memory size will approximately be 983,040 bits. This is a very large memory to be implemented. To reduce this size, a better traceback implementation is done that only encodes all the parameters said before to a single bit that tells whether the branch that is connecting the two states together is an upper or lower branch then with a simple left shift operation, we can figure out the previous state simply as shown in the figure.

The figure shows an example of the used traceback operation method where the current state is 23 and the previous state is required to complete tracing back to the initial state. The available two previous states are 46 and 47 only according to the trellis diagram of the encoder. The control unit reads the bit stored in the location of the path record memory that is [t+1][23] and then performs a shift left operation to the current value 23 then finally adds the stored bit read from the memory. If the saved bit was 0', the previous state will be 46. And if the saved bit was 1', the previous state will be 47. In conclusion, by saving only one bit, we could perform the traceback operation correctly without having a very large memory.

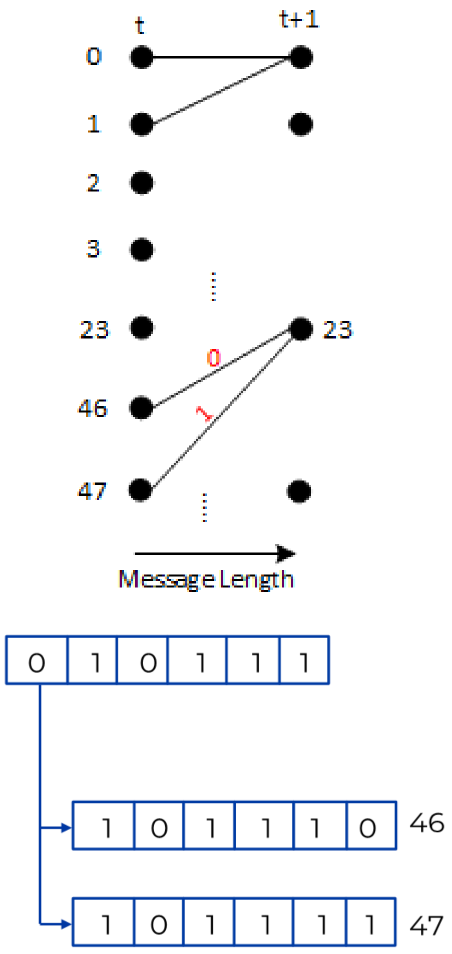


Figure [101]: Another implementation of traceback operation to reduce memory size

3.15.4. MATLAB Results

High Level Implementation of the algorithm was done on MATLAB and was tested with a random vector with the maximum transport block size (2560). The figure below shows the different error counts against different SNR values.

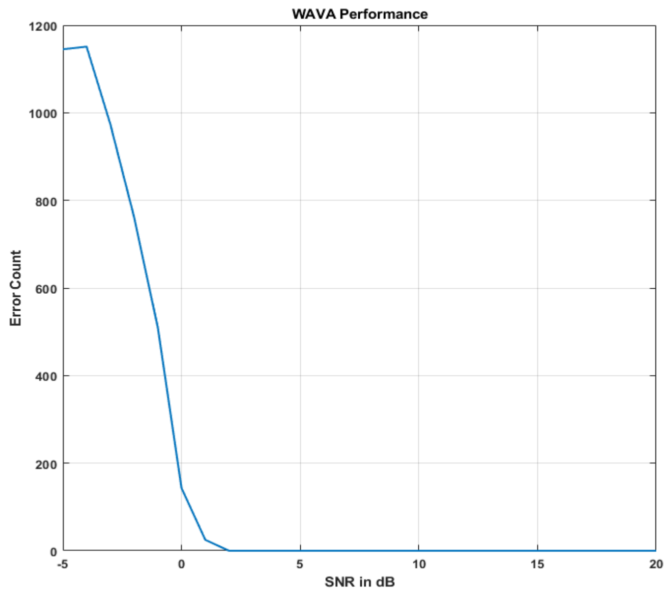
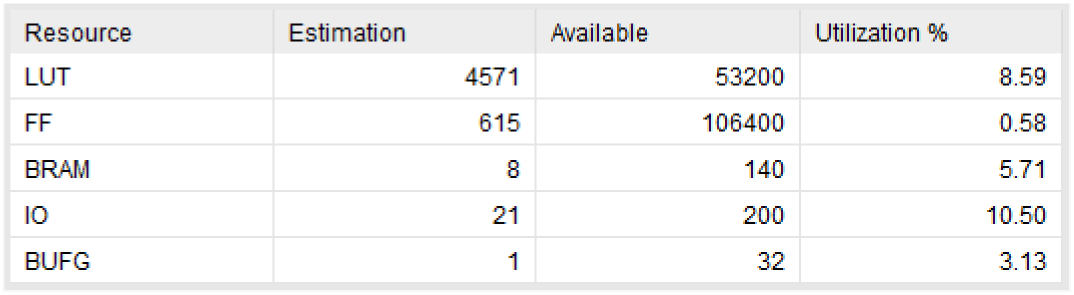


Figure [102]: MATLAB Implementation of WAVA Results

3.15.5. Synthesis Results

The Figure below shows the overall synthesized area in FPGA cells on the targeted FPGA. Synthesis is done using Vivado Design Suite which includes block's utilization in resources (LUTs, FFs, DSPs, etc.)



Resource	Estimation	Available	Utilization %
LUT	4571	53200	8.59
FF	615	106400	0.58
BRAM	8	140	5.71
IO	21	200	10.50
BUFG	1	32	3.13

Figure [103]: Viterbi Decoder Synthesis Utilization

3.16. Cyclic Redundancy Check

3.16.1. Top Module

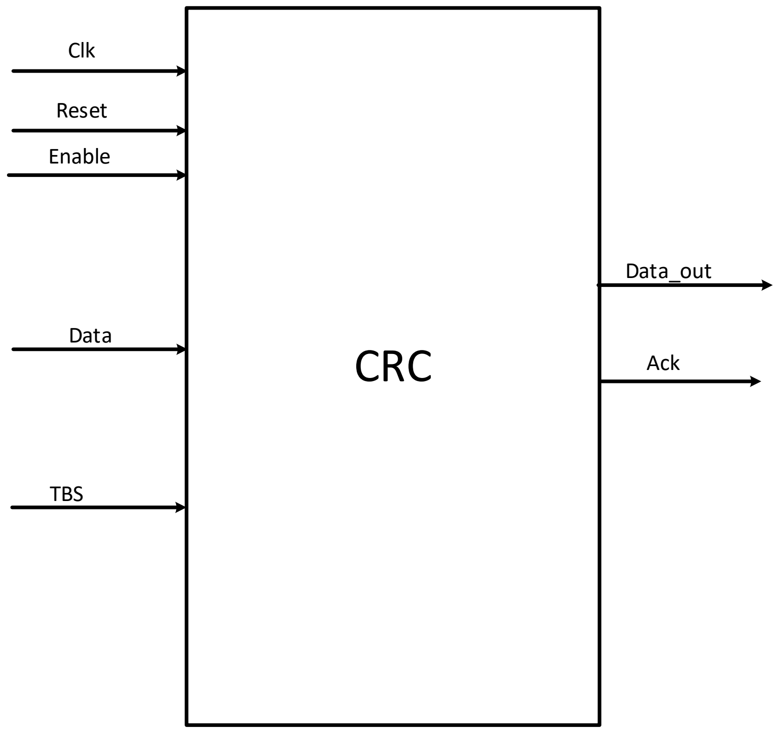


Figure [104]: CRC Top Module

Table [25]: CRC Interface Table

Signal Name	**Direction**	**Width**	**Description**
Clk_260	Input	1	3.84 MHz clock signal
Reset	Input	1	Global reset signal
Enable	Input	1	Enable signal for the Demodulator module to start operating
Data	Input	1	Data received from the decoder
TBS	Input	12	Transport Block Size of the incoming data
Data_out	Output	1	Serial data bits output
Ack	Output	1	Ack to indicate data validity

3.16.2. Detailed Hardware

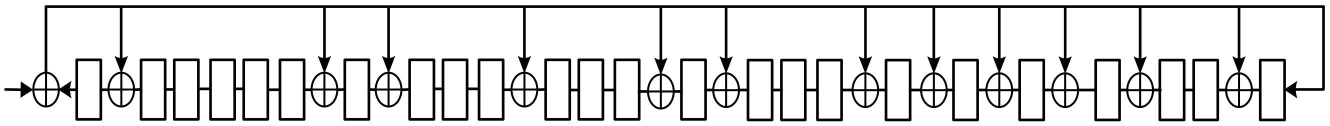


Figure [105]: CRC Detailed Design

Description:

The CRC value is generated by XORing the input bit with the value of the final register and the result is used as feedback to all the XORs implementing the polynomial. This implementation reduces the number of cycles taken to perform the check.

The Register at first is initialized to 0 then after passing the received data bitstream of length equal TBS+ 24 bits CRC the register should contain Zeros once again and that's when the acknowledge signal is asserted.

3.16.3. Results

3.16.3.1. MATLAB vs. RTL

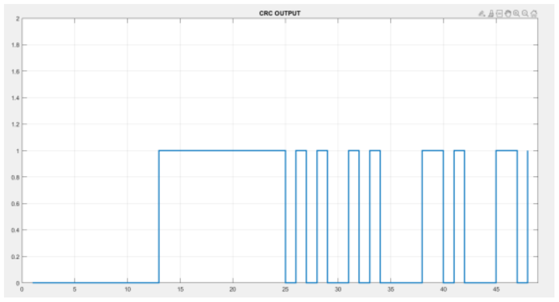


Figure [106]: CRC MATLAB Results

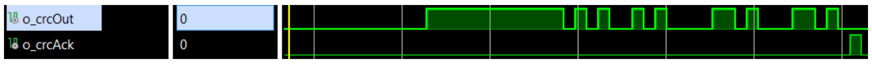
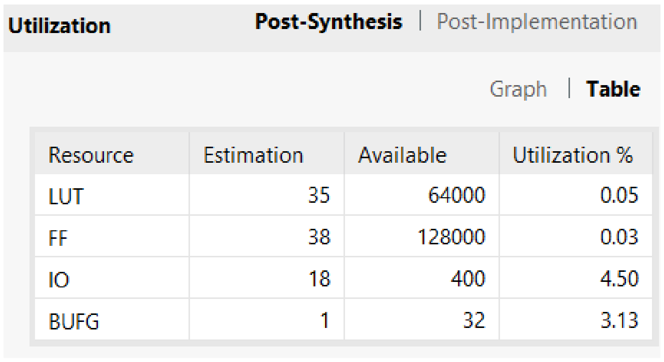


Figure [107]: CRC RTL Results

3.16.3.2. Synthesis Results



Utilization **Post-Synthesis** | Post-Implementation

Graph | **Table**

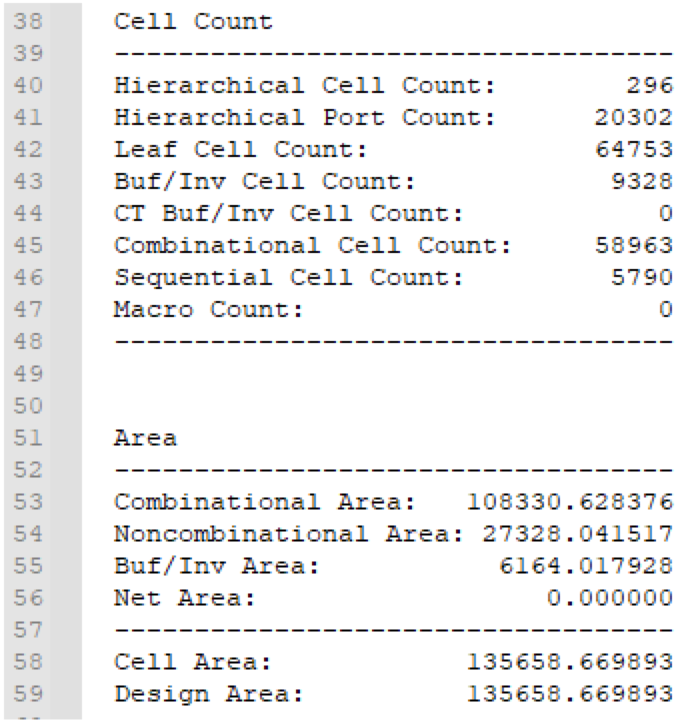
Resource	Estimation	Available	Utilization %
LUT	35	64000	0.05
FF	38	128000	0.03
IO	18	400	4.50
BUFG	1	32	3.13

Figure [108]: CRC Synthesis Results

4.1. Synopsys Design Compiler Synthesis Results

4.1.1. Coarse Synchronizer

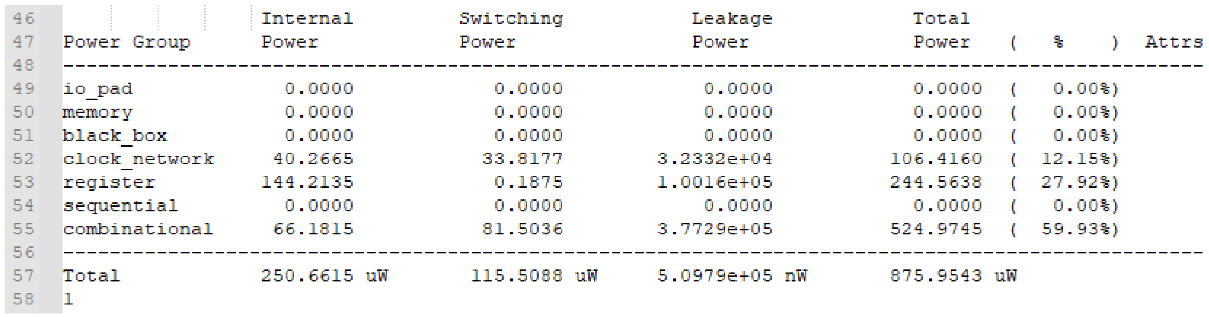
4.1.1.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:        296
Hierarchical Port Count:      20302
Leaf Cell Count:              64753
Buf/Inv Cell Count:            9328
CT Buf/Inv Cell Count:            0
Combinational Cell Count:     58963
Sequential Cell Count:         5790
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:   108330.628376
Noncombinational Area: 27328.041517
Buf/Inv Area:           6164.017928
Net Area:                  0.000000
-----------------------------------
Cell Area:            135658.669893
Design Area:          135658.669893
```

Figure [109]: Coarse Synchronizer DC Area Results

4.1.1.2. Power Results

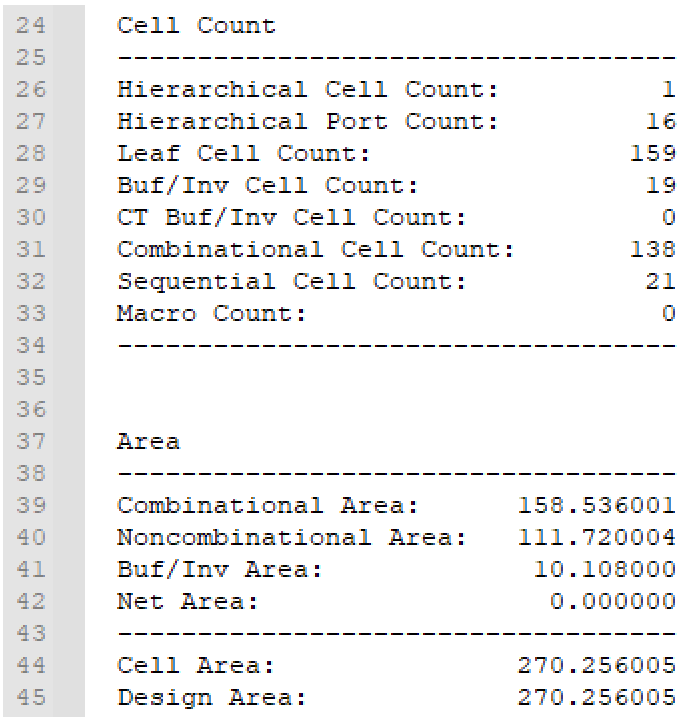


```
                    Internal         Switching           Leakage            Total
Power Group         Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad                0.0000           0.0000            0.0000             0.0000  (   0.00%)
memory                0.0000           0.0000            0.0000             0.0000  (   0.00%)
black_box             0.0000           0.0000            0.0000             0.0000  (   0.00%)
clock_network        40.2665          33.8177        3.2332e+04           106.4160  (  12.15%)
register            144.2135           0.1875        1.0016e+05           244.5638  (  27.92%)
sequential            0.0000           0.0000            0.0000             0.0000  (   0.00%)
combinational        66.1815          81.5036        3.7729e+05           524.9745  (  59.93%)
--------------------------------------------------------------------------------------------------
Total               250.6615 uW      115.5088 uW     5.0979e+05 nW      875.9543 uW
1
```

Figure [110]: Coarse Synchronizer DC Power Results

4.1.2. CP Remover and Downsampler

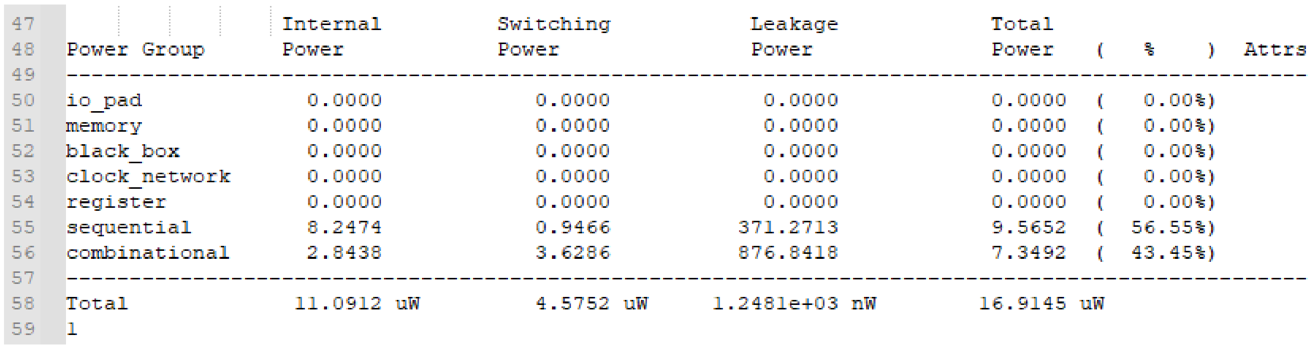
4.1.2.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:          1
Hierarchical Port Count:         16
Leaf Cell Count:                159
Buf/Inv Cell Count:              19
CT Buf/Inv Cell Count:            0
Combinational Cell Count:       138
Sequential Cell Count:           21
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:      158.536001
Noncombinational Area:   111.720004
Buf/Inv Area:             10.108000
Net Area:                  0.000000
-----------------------------------
Cell Area:               270.256005
Design Area:             270.256005
```

Figure [111]: CP Remover and Downsampler DC Area Results

4.1.2.2. Power Results

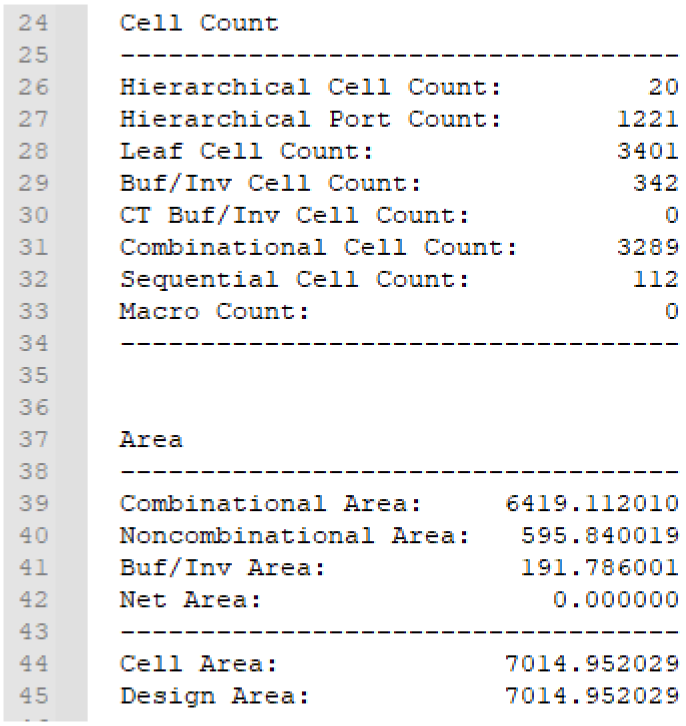


```
                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000           0.0000             0.0000             0.0000  (   0.00%)
memory             0.0000           0.0000             0.0000             0.0000  (   0.00%)
black_box          0.0000           0.0000             0.0000             0.0000  (   0.00%)
clock_network      0.0000           0.0000             0.0000             0.0000  (   0.00%)
register           0.0000           0.0000             0.0000             0.0000  (   0.00%)
sequential         8.2474           0.9466           371.2713             9.5652  (  56.55%)
combinational      2.8438           3.6286           876.8418             7.3492  (  43.45%)
--------------------------------------------------------------------------------------------------
Total             11.0912 uW        4.5752 uW     1.2481e+03 nW        16.9145 uW
1
```

Figure [112]: CP Remover and Downsampler DC Power Results

4.1.3. CFO Corrector

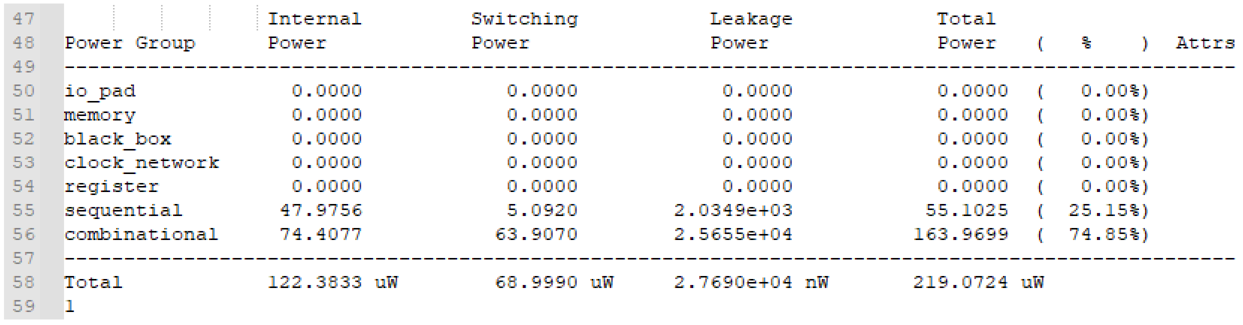
4.1.3.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:         20
Hierarchical Port Count:       1221
Leaf Cell Count:               3401
Buf/Inv Cell Count:             342
CT Buf/Inv Cell Count:            0
Combinational Cell Count:      3289
Sequential Cell Count:          112
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:     6419.112010
Noncombinational Area:   595.840019
Buf/Inv Area:            191.786001
Net Area:                  0.000000
-----------------------------------
Cell Area:              7014.952029
Design Area:            7014.952029
```

Figure [113]: CFO Corrector DC Area Results

4.1.3.2. Power Results

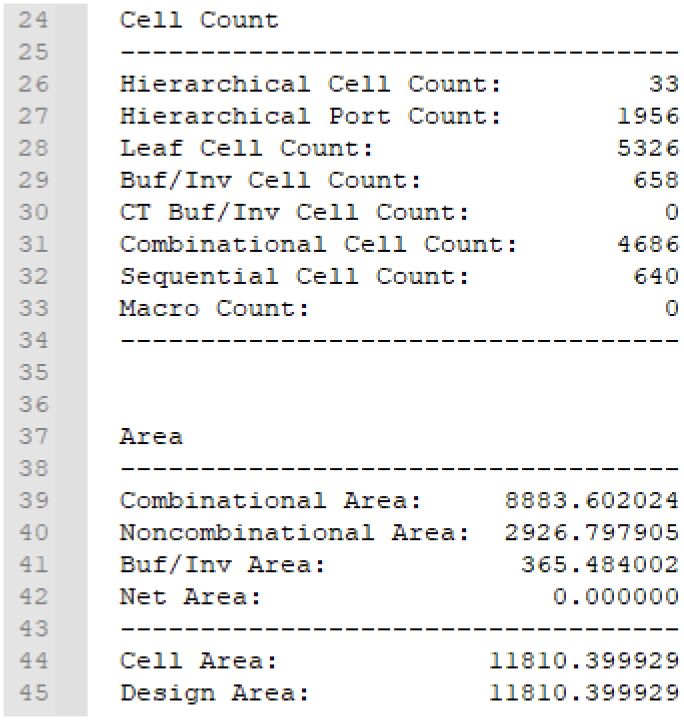


```
                         Internal         Switching           Leakage            Total
Power Group              Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------------
io_pad                      0.0000           0.0000            0.0000            0.0000  (   0.00%)
memory                      0.0000           0.0000            0.0000            0.0000  (   0.00%)
black_box                   0.0000           0.0000            0.0000            0.0000  (   0.00%)
clock_network               0.0000           0.0000            0.0000            0.0000  (   0.00%)
register                    0.0000           0.0000            0.0000            0.0000  (   0.00%)
sequential                 47.9756           5.0920        2.0349e+03           55.1025  (  25.15%)
combinational              74.4077          63.9070        2.5655e+04          163.9699  (  74.85%)
--------------------------------------------------------------------------------------------------------
Total                    122.3833 uW        68.9990 uW     2.7690e+04 nW       219.0724 uW
1
```

Figure [114]: CFO Corrector DC Power Results

4.1.4. FFT Engine

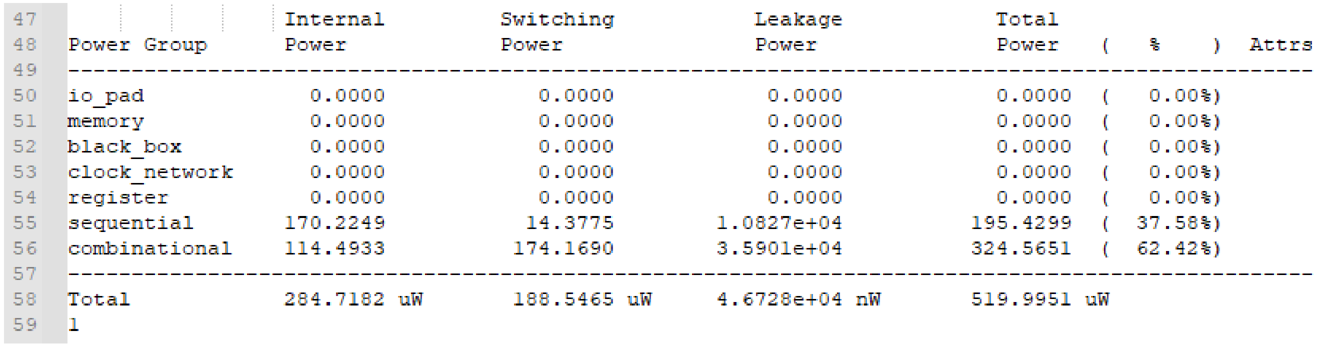
4.1.4.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:         33
Hierarchical Port Count:       1956
Leaf Cell Count:               5326
Buf/Inv Cell Count:             658
CT Buf/Inv Cell Count:            0
Combinational Cell Count:      4686
Sequential Cell Count:          640
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:     8883.602024
Noncombinational Area:  2926.797905
Buf/Inv Area:            365.484002
Net Area:                  0.000000
-----------------------------------
Cell Area:             11810.399929
Design Area:           11810.399929
```

Figure [115]: FFT Engine DC Area Results

4.1.4.2. Power Results

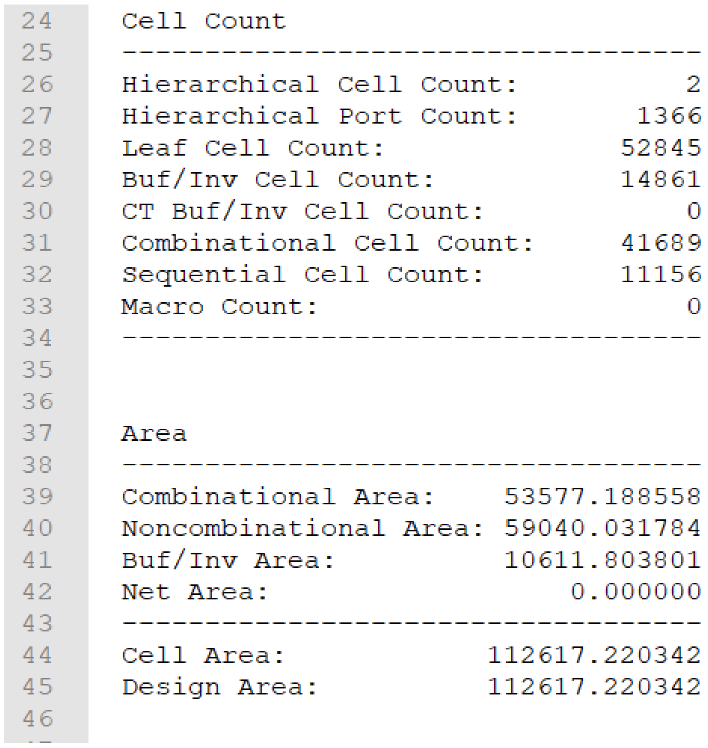


```
                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000           0.0000             0.0000            0.0000  (   0.00%)
memory             0.0000           0.0000             0.0000            0.0000  (   0.00%)
black_box          0.0000           0.0000             0.0000            0.0000  (   0.00%)
clock_network      0.0000           0.0000             0.0000            0.0000  (   0.00%)
register           0.0000           0.0000             0.0000            0.0000  (   0.00%)
sequential       170.2249          14.3775         1.0827e+04          195.4299  (  37.58%)
combinational    114.4933         174.1690         3.5901e+04          324.5651  (  62.42%)
--------------------------------------------------------------------------------------------------
Total            284.7182 uW      188.5465 uW      4.6728e+04 nW       519.9951 uW
1
```

Figure [116]: FFT Engine DC Power Results

4.1.5. Resource Demapper

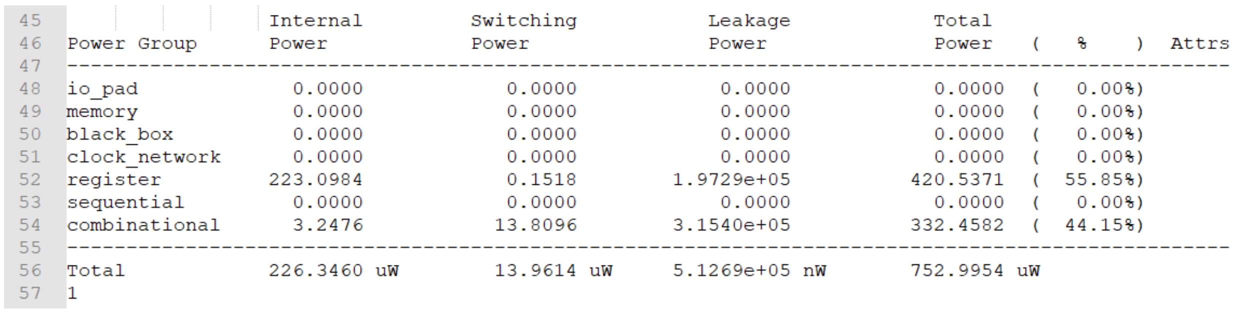
4.1.5.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:          2
Hierarchical Port Count:       1366
Leaf Cell Count:              52845
Buf/Inv Cell Count:           14861
CT Buf/Inv Cell Count:            0
Combinational Cell Count:     41689
Sequential Cell Count:        11156
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:    53577.188558
Noncombinational Area: 59040.031784
Buf/Inv Area:          10611.803801
Net Area:                  0.000000
-----------------------------------
Cell Area:            112617.220342
Design Area:          112617.220342
```

Figure [117]: Resource Demapper Area report

4.1.5.2. Power Results

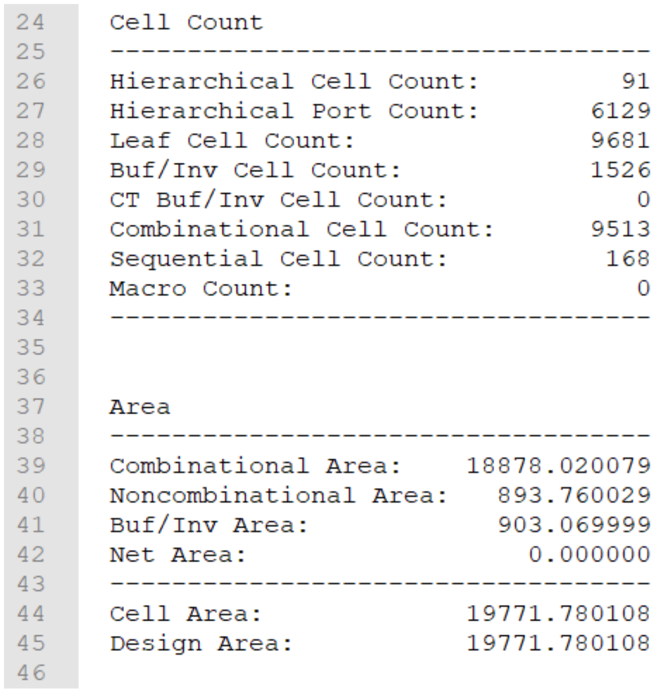


```
                    Internal          Switching          Leakage           Total
Power Group         Power             Power              Power             Power     (   %    )  Attrs
-----------------------------------------------------------------------------------------------------
io_pad                0.0000            0.0000             0.0000            0.0000  (   0.00%)
memory                0.0000            0.0000             0.0000            0.0000  (   0.00%)
black_box             0.0000            0.0000             0.0000            0.0000  (   0.00%)
clock_network         0.0000            0.0000             0.0000            0.0000  (   0.00%)
register            223.0984            0.1518         1.9729e+05          420.5371  (  55.85%)
sequential            0.0000            0.0000             0.0000            0.0000  (   0.00%)
combinational         3.2476           13.8096         3.1540e+05          332.4582  (  44.15%)
-----------------------------------------------------------------------------------------------------
Total               226.3460 uW        13.9614 uW      5.1269e+05 nW       752.9954 uW
1
```

Figure [118]: Resource Demapper Power report

4.1.6. Channel Estimation

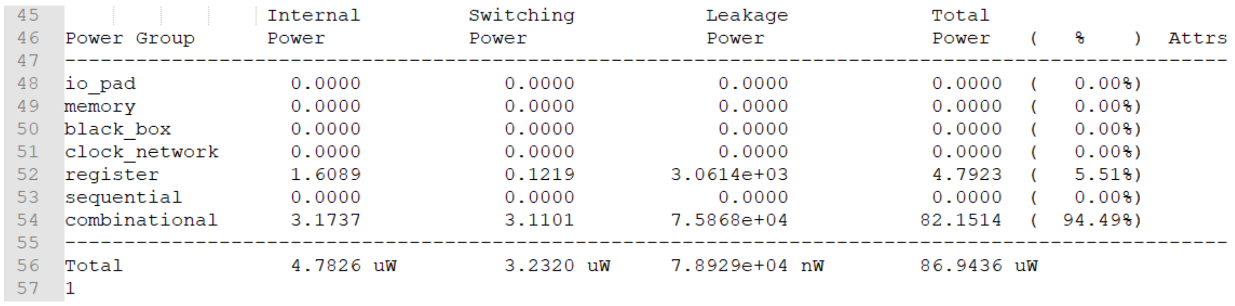
4.1.6.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:         91
Hierarchical Port Count:       6129
Leaf Cell Count:               9681
Buf/Inv Cell Count:            1526
CT Buf/Inv Cell Count:            0
Combinational Cell Count:      9513
Sequential Cell Count:          168
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:    18878.020079
Noncombinational Area:   893.760029
Buf/Inv Area:            903.069999
Net Area:                  0.000000
-----------------------------------
Cell Area:             19771.780108
Design Area:           19771.780108
```

Figure [119]: Channel Estimation Area report

4.1.6.2. Power Results

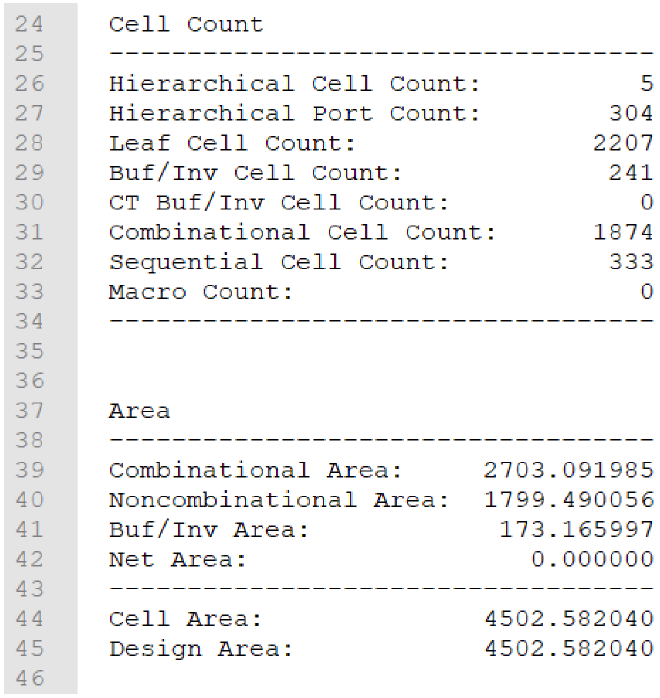


```
                  Internal         Switching           Leakage            Total
Power Group       Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad              0.0000           0.0000            0.0000            0.0000  (   0.00%)
memory              0.0000           0.0000            0.0000            0.0000  (   0.00%)
black_box           0.0000           0.0000            0.0000            0.0000  (   0.00%)
clock_network       0.0000           0.0000            0.0000            0.0000  (   0.00%)
register            1.6089           0.1219        3.0614e+03            4.7923  (   5.51%)
sequential          0.0000           0.0000            0.0000            0.0000  (   0.00%)
combinational       3.1737           3.1101        7.5868e+04           82.1514  (  94.49%)
--------------------------------------------------------------------------------------------------
Total               4.7826 uW        3.2320 uW     7.8929e+04 nW       86.9436 uW
1
```

Figure [120]: Channel Estimation Power report

4.1.7. NRS Values Generator

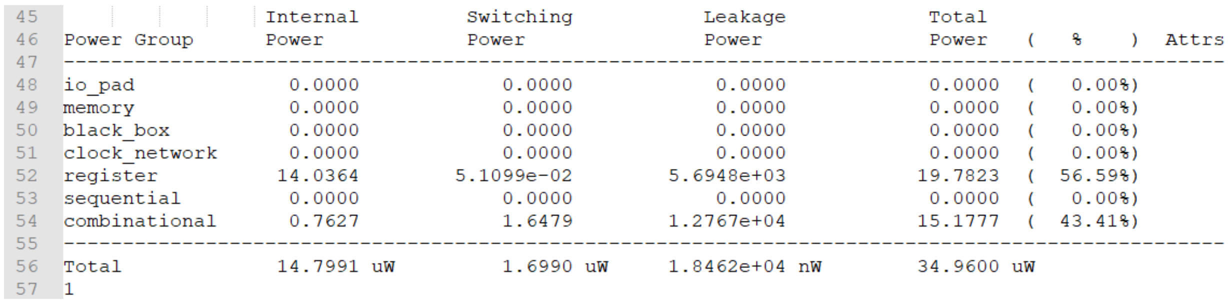
4.1.7.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:          5
Hierarchical Port Count:        304
Leaf Cell Count:               2207
Buf/Inv Cell Count:             241
CT Buf/Inv Cell Count:            0
Combinational Cell Count:      1874
Sequential Cell Count:          333
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:     2703.091985
Noncombinational Area:  1799.490056
Buf/Inv Area:            173.165997
Net Area:                  0.000000
-----------------------------------
Cell Area:              4502.582040
Design Area:            4502.582040
```

Figure [121]: NRS Values Generator Area report

4.1.7.2. Power Results

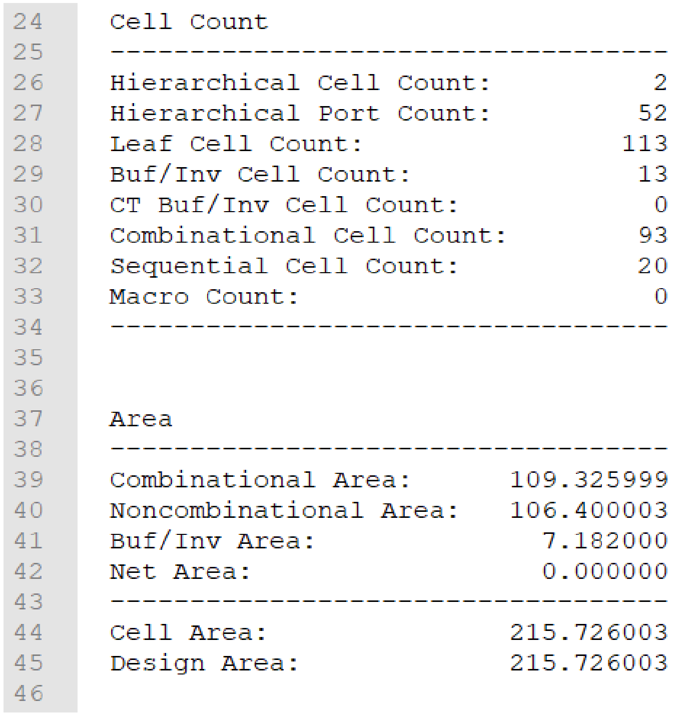


```
                  Internal         Switching           Leakage            Total
Power Group       Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------------
io_pad              0.0000           0.0000              0.0000             0.0000  (   0.00%)
memory              0.0000           0.0000              0.0000             0.0000  (   0.00%)
black_box           0.0000           0.0000              0.0000             0.0000  (   0.00%)
clock_network       0.0000           0.0000              0.0000             0.0000  (   0.00%)
register           14.0364        5.1099e-02         5.6948e+03            19.7823  (  56.59%)
sequential          0.0000           0.0000              0.0000             0.0000  (   0.00%)
combinational       0.7627           1.6479          1.2767e+04            15.1777  (  43.41%)
--------------------------------------------------------------------------------------------------------
Total              14.7991 uW        1.6990 uW     1.8462e+04 nW           34.9600 uW
1
```

Figure [122]: NRS Values Generator Power report

4.1.8. NRS Index Generator

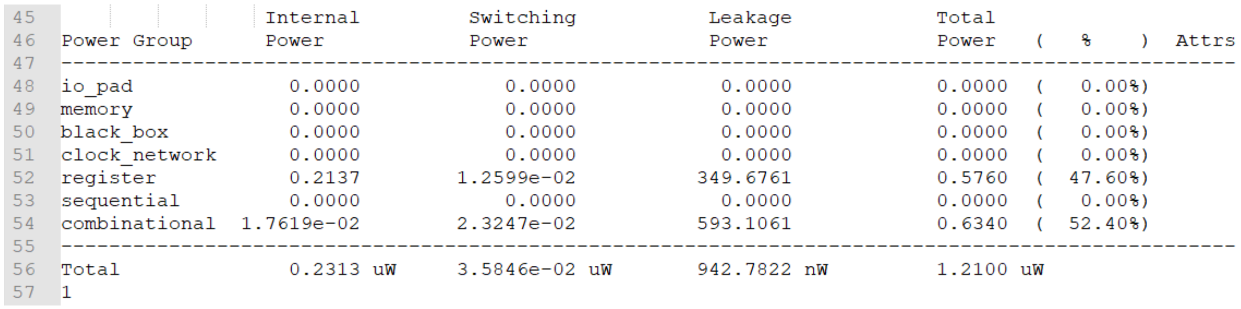
4.1.8.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:          2
Hierarchical Port Count:         52
Leaf Cell Count:                113
Buf/Inv Cell Count:              13
CT Buf/Inv Cell Count:            0
Combinational Cell Count:        93
Sequential Cell Count:           20
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:      109.325999
Noncombinational Area:   106.400003
Buf/Inv Area:              7.182000
Net Area:                  0.000000
-----------------------------------
Cell Area:               215.726003
Design Area:             215.726003
```

Figure [123]: NRS Index Generator Area report

4.1.8.2. Power Results

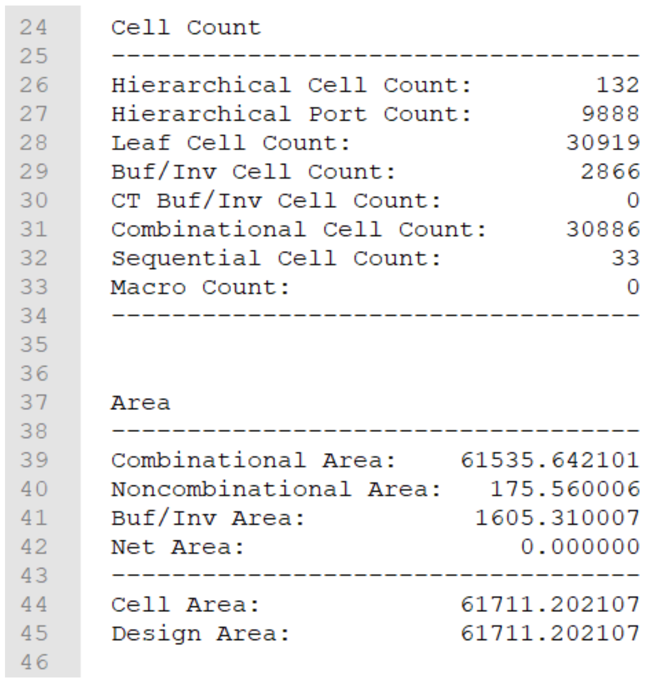


```
                        Internal         Switching           Leakage            Total
Power Group             Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------------
io_pad                    0.0000            0.0000            0.0000            0.0000  (   0.00%)
memory                    0.0000            0.0000            0.0000            0.0000  (   0.00%)
black_box                 0.0000            0.0000            0.0000            0.0000  (   0.00%)
clock_network             0.0000            0.0000            0.0000            0.0000  (   0.00%)
register                  0.2137        1.2599e-02          349.6761            0.5760  (  47.60%)
sequential                0.0000            0.0000            0.0000            0.0000  (   0.00%)
combinational         1.7619e-02        2.3247e-02          593.1061            0.6340  (  52.40%)
--------------------------------------------------------------------------------------------------------
Total                     0.2313 uW     3.5846e-02 uW       942.7822 nW         1.2100 uW
1
```

Figure [124]: NRS Index Generator Power report

4.1.9. Channel Equalizer

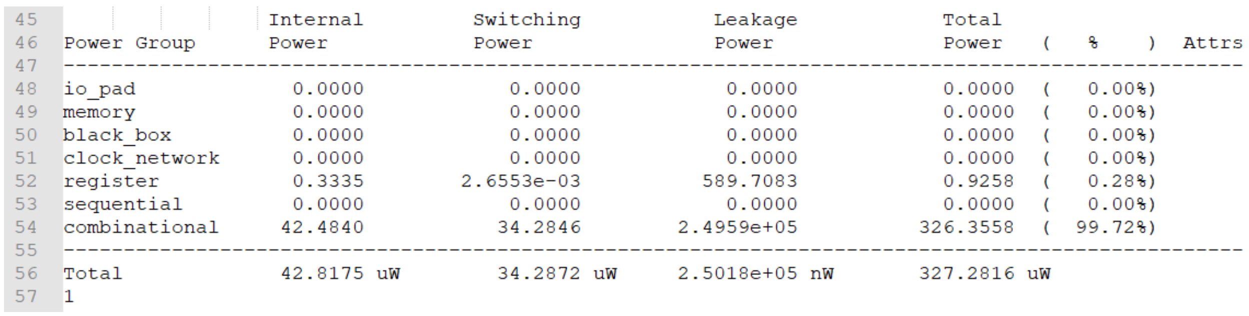
4.1.9.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:        132
Hierarchical Port Count:       9888
Leaf Cell Count:              30919
Buf/Inv Cell Count:            2866
CT Buf/Inv Cell Count:            0
Combinational Cell Count:     30886
Sequential Cell Count:           33
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:    61535.642101
Noncombinational Area:   175.560006
Buf/Inv Area:           1605.310007
Net Area:                  0.000000
-----------------------------------
Cell Area:             61711.202107
Design Area:           61711.202107
```

Figure [125]: Channel Equalizer Area report

4.1.9.2. Power Results

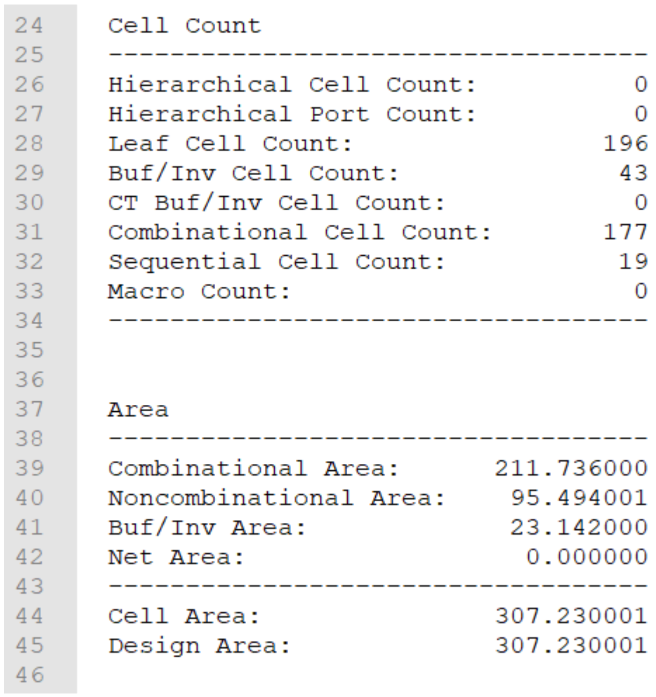


```
                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power    (    %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000           0.0000              0.0000             0.0000  (   0.00%)
memory             0.0000           0.0000              0.0000             0.0000  (   0.00%)
black_box          0.0000           0.0000              0.0000             0.0000  (   0.00%)
clock_network      0.0000           0.0000              0.0000             0.0000  (   0.00%)
register           0.3335        2.6553e-03           589.7083             0.9258  (   0.28%)
sequential         0.0000           0.0000              0.0000             0.0000  (   0.00%)
combinational     42.4840          34.2846          2.4959e+05           326.3558  (  99.72%)
--------------------------------------------------------------------------------------------------
Total             42.8175 uW       34.2872 uW       2.5018e+05 nW      327.2816 uW
1
```

Figure [126]: Channel Equalizer Power report

4.1.10. Parallel to Serial and NRS removal

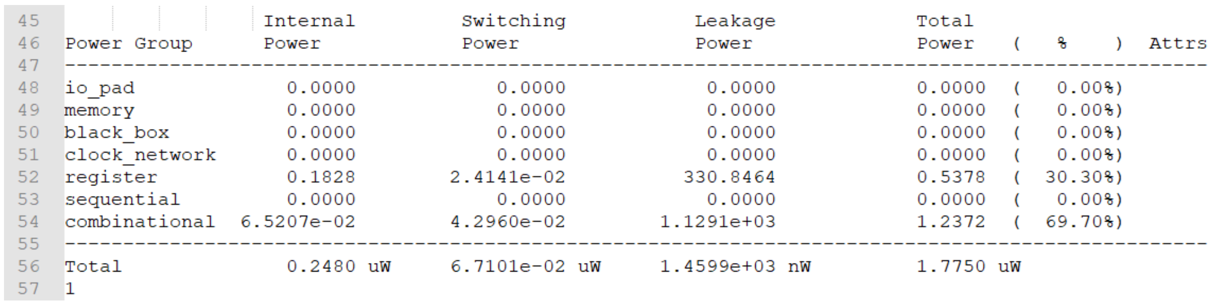
4.1.10.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:          0
Hierarchical Port Count:          0
Leaf Cell Count:                196
Buf/Inv Cell Count:              43
CT Buf/Inv Cell Count:            0
Combinational Cell Count:       177
Sequential Cell Count:           19
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:      211.736000
Noncombinational Area:    95.494001
Buf/Inv Area:             23.142000
Net Area:                  0.000000
-----------------------------------
Cell Area:               307.230001
Design Area:             307.230001
```

Figure [127]: Parallel to Serial and NRS removal Area report

4.1.10.2. Power Results

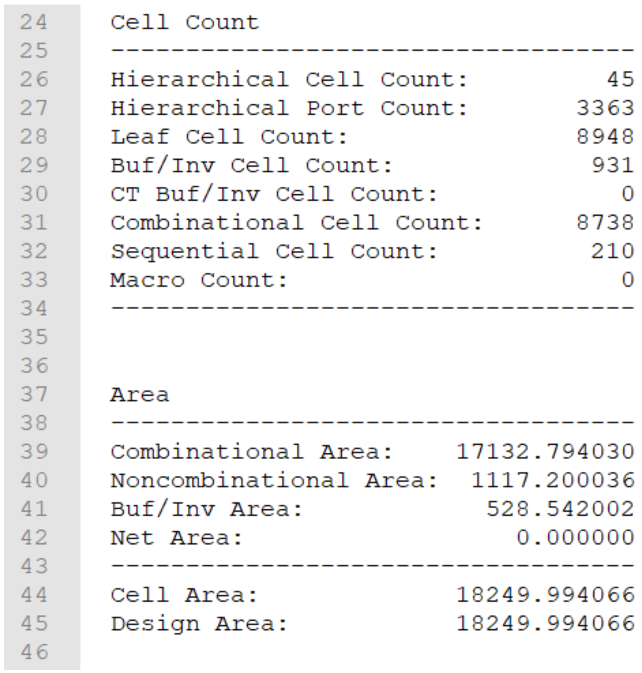


```
                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power    (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000            0.0000            0.0000            0.0000  (   0.00%)
memory             0.0000            0.0000            0.0000            0.0000  (   0.00%)
black_box          0.0000            0.0000            0.0000            0.0000  (   0.00%)
clock_network      0.0000            0.0000            0.0000            0.0000  (   0.00%)
register           0.1828        2.4141e-02          330.8464            0.5378  (  30.30%)
sequential         0.0000            0.0000            0.0000            0.0000  (   0.00%)
combinational  6.5207e-02        4.2960e-02        1.1291e+03            1.2372  (  69.70%)
--------------------------------------------------------------------------------------------------
Total              0.2480 uW     6.7101e-02 uW     1.4599e+03 nW         1.7750 uW
1
```

Figure [128]: Parallel to Serial and NRS removal Power report

4.1.11. Fine Synchronizer

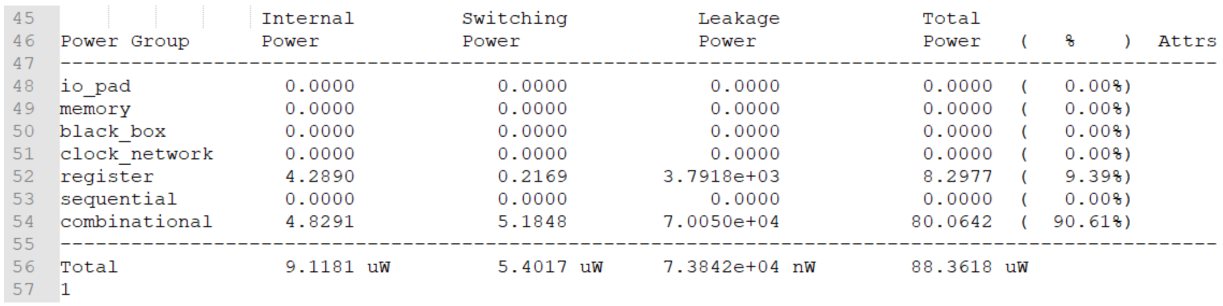
4.1.11.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:         45
Hierarchical Port Count:       3363
Leaf Cell Count:               8948
Buf/Inv Cell Count:             931
CT Buf/Inv Cell Count:            0
Combinational Cell Count:      8738
Sequential Cell Count:          210
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:    17132.794030
Noncombinational Area:  1117.200036
Buf/Inv Area:            528.542002
Net Area:                  0.000000
-----------------------------------
Cell Area:             18249.994066
Design Area:           18249.994066

```

Figure [129]: Fine Synchronization Area Report

4.1.11.2. Power Results

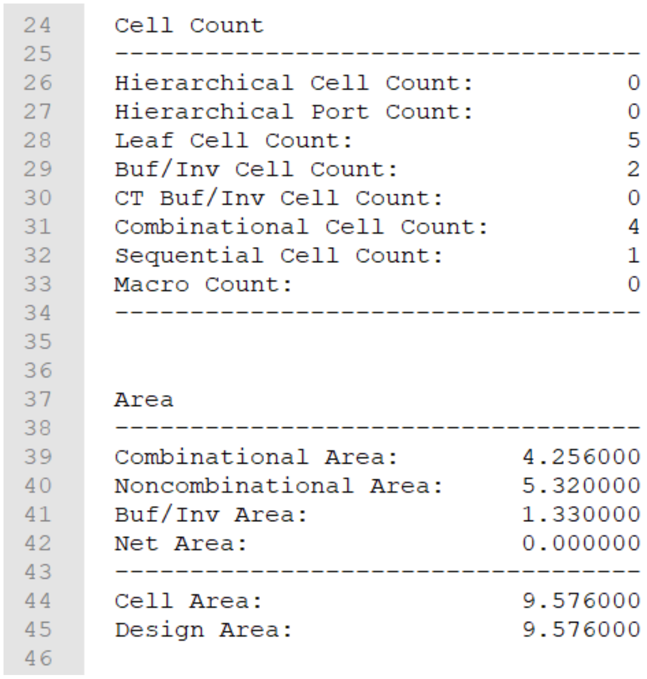


```
                        Internal          Switching           Leakage            Total
Power Group             Power             Power               Power              Power   (   %    )  Attrs
------------------------------------------------------------------------------------------------------------
io_pad                     0.0000            0.0000            0.0000            0.0000  (   0.00%)
memory                     0.0000            0.0000            0.0000            0.0000  (   0.00%)
black_box                  0.0000            0.0000            0.0000            0.0000  (   0.00%)
clock_network              0.0000            0.0000            0.0000            0.0000  (   0.00%)
register                   4.2890            0.2169        3.7918e+03            8.2977  (   9.39%)
sequential                 0.0000            0.0000            0.0000            0.0000  (   0.00%)
combinational              4.8291            5.1848        7.0050e+04           80.0642  (  90.61%)
------------------------------------------------------------------------------------------------------------
Total                      9.1181 uW         5.4017 uW     7.3842e+04 nW       88.3618 uW
1
```

Figure [130]: Fine Synchronization Power Report

4.1.12. Demodulator

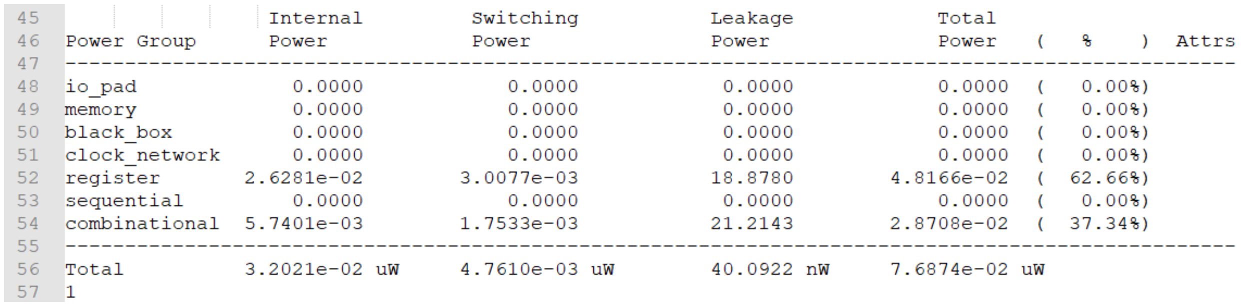
4.1.12.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:          0
Hierarchical Port Count:          0
Leaf Cell Count:                  5
Buf/Inv Cell Count:               2
CT Buf/Inv Cell Count:            0
Combinational Cell Count:         4
Sequential Cell Count:            1
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:        4.256000
Noncombinational Area:     5.320000
Buf/Inv Area:              1.330000
Net Area:                  0.000000
-----------------------------------
Cell Area:                 9.576000
Design Area:               9.576000
```

Figure [131]: Demodulator Area Report

4.1.12.2. Power Results

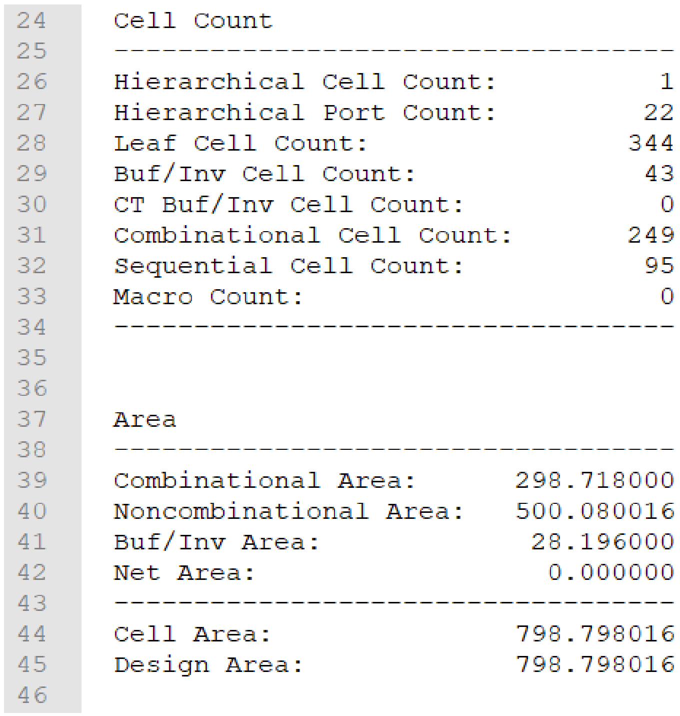


```
                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000           0.0000            0.0000             0.0000  (   0.00%)
memory             0.0000           0.0000            0.0000             0.0000  (   0.00%)
black_box          0.0000           0.0000            0.0000             0.0000  (   0.00%)
clock_network      0.0000           0.0000            0.0000             0.0000  (   0.00%)
register       2.6281e-02       3.0077e-03           18.8780         4.8166e-02  (  62.66%)
sequential         0.0000           0.0000            0.0000             0.0000  (   0.00%)
combinational  5.7401e-03       1.7533e-03           21.2143         2.8708e-02  (  37.34%)
--------------------------------------------------------------------------------------------------
Total          3.2021e-02 uW    4.7610e-03 uW        40.0922 nW      7.6874e-02 uW
1
```

Figure [132]: Demodulator Power Report

4.1.13. Descrambler

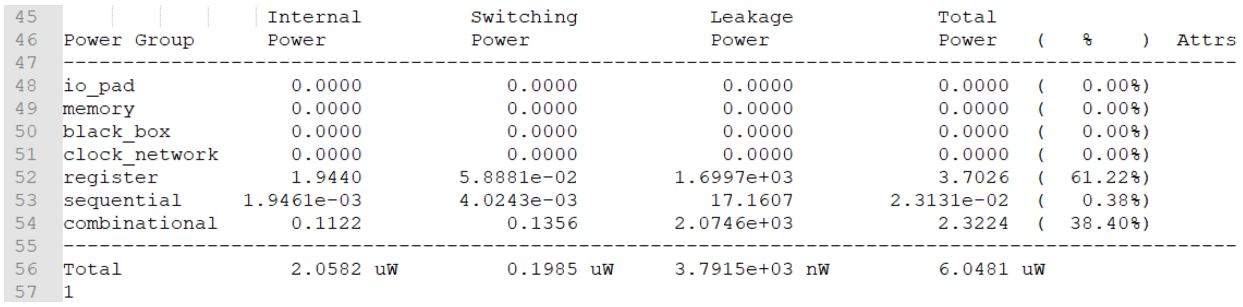
4.1.13.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:          1
Hierarchical Port Count:         22
Leaf Cell Count:                344
Buf/Inv Cell Count:              43
CT Buf/Inv Cell Count:            0
Combinational Cell Count:       249
Sequential Cell Count:           95
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:      298.718000
Noncombinational Area:   500.080016
Buf/Inv Area:             28.196000
Net Area:                  0.000000
-----------------------------------
Cell Area:               798.798016
Design Area:             798.798016
```

Figure [133]: Descrambler Area Report

4.1.13.2. Power Results

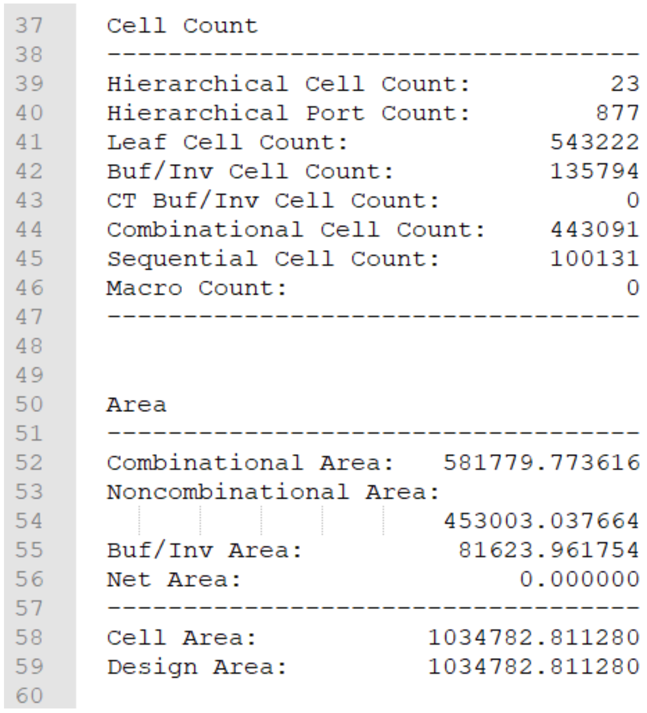


```
                  Internal         Switching           Leakage            Total
Power Group       Power            Power               Power              Power    (   %    )  Attrs
--------------------------------------------------------------------------------------------------------
io_pad              0.0000            0.0000            0.0000            0.0000  (   0.00%)
memory              0.0000            0.0000            0.0000            0.0000  (   0.00%)
black_box           0.0000            0.0000            0.0000            0.0000  (   0.00%)
clock_network       0.0000            0.0000            0.0000            0.0000  (   0.00%)
register            1.9440        5.8881e-02        1.6997e+03            3.7026  (  61.22%)
sequential      1.9461e-03        4.0243e-03           17.1607        2.3131e-02  (   0.38%)
combinational       0.1122            0.1356        2.0746e+03            2.3224  (  38.40%)
--------------------------------------------------------------------------------------------------------
Total               2.0582 uW         0.1985 uW     3.7915e+03 nW         6.0481 uW
1
```

Figure [134]: Descrambler Power Report

4.1.14. Rate De-Matcher

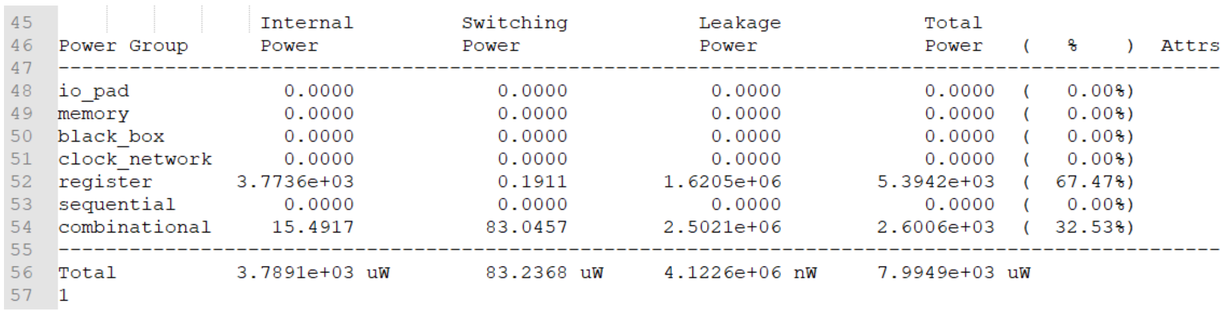
4.1.14.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:         23
Hierarchical Port Count:        877
Leaf Cell Count:             543222
Buf/Inv Cell Count:          135794
CT Buf/Inv Cell Count:            0
Combinational Cell Count:    443091
Sequential Cell Count:       100131
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:   581779.773616
Noncombinational Area:
                      453003.037664
Buf/Inv Area:          81623.961754
Net Area:                  0.000000
-----------------------------------
Cell Area:           1034782.811280
Design Area:         1034782.811280
```

Figure [135]: Rate De-Matcher Area Report

4.1.14.2. Power Results

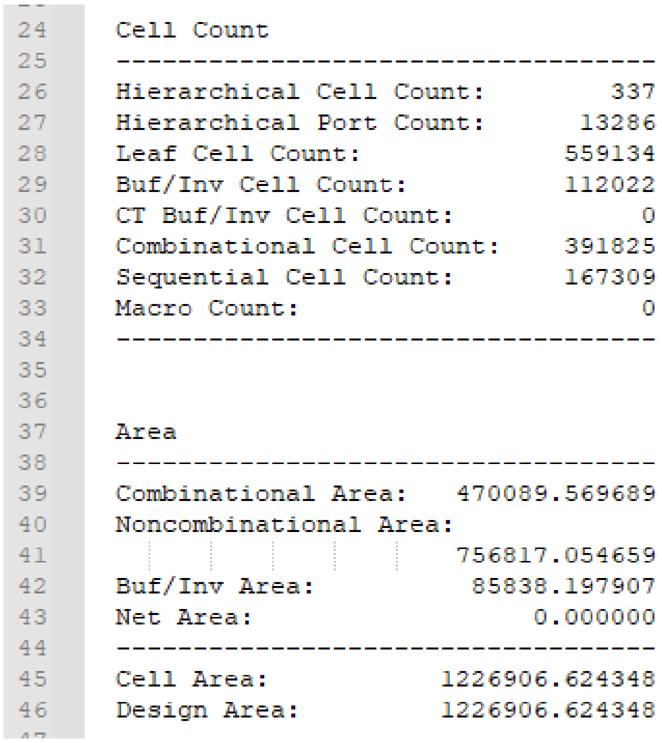


```
                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power    (   %    )  Attrs
-------------------------------------------------------------------------------------------------
io_pad             0.0000           0.0000             0.0000             0.0000  (   0.00%)
memory             0.0000           0.0000             0.0000             0.0000  (   0.00%)
black_box          0.0000           0.0000             0.0000             0.0000  (   0.00%)
clock_network      0.0000           0.0000             0.0000             0.0000  (   0.00%)
register         3.7736e+03         0.1911          1.6205e+06         5.3942e+03  (  67.47%)
sequential         0.0000           0.0000             0.0000             0.0000  (   0.00%)
combinational     15.4917          83.0457          2.5021e+06         2.6006e+03  (  32.53%)
-------------------------------------------------------------------------------------------------
Total            3.7891e+03 uW     83.2368 uW     4.1226e+06 nW     7.9949e+03 uW
1
```

Figure [136]: Rate De-Matcher Power Report

4.1.15. Viterbi Decoder

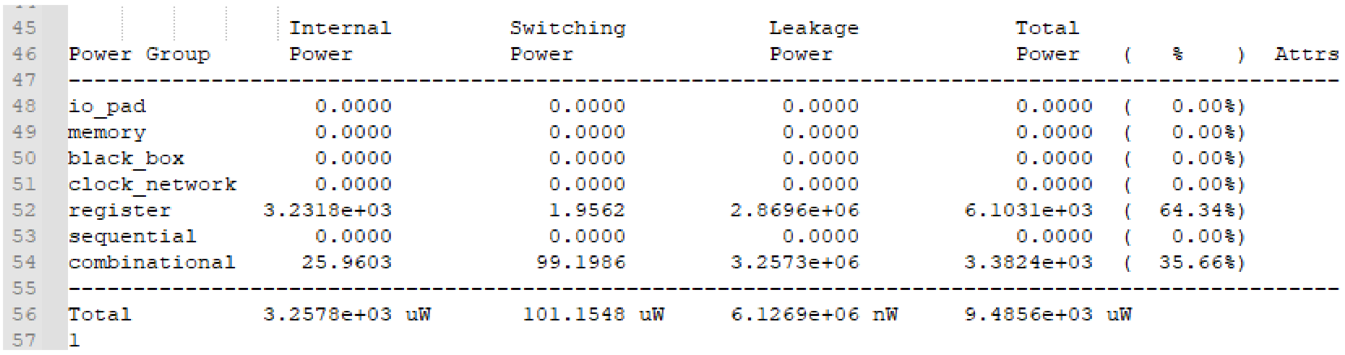
4.1.15.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:        337
Hierarchical Port Count:      13286
Leaf Cell Count:             559134
Buf/Inv Cell Count:          112022
CT Buf/Inv Cell Count:            0
Combinational Cell Count:    391825
Sequential Cell Count:       167309
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:   470089.569689
Noncombinational Area:
                      756817.054659
Buf/Inv Area:          85838.197907
Net Area:                  0.000000
-----------------------------------
Cell Area:           1226906.624348
Design Area:         1226906.624348
```

Figure [137]: Viterbi Decoder Area Reports

4.1.15.2. Power Results

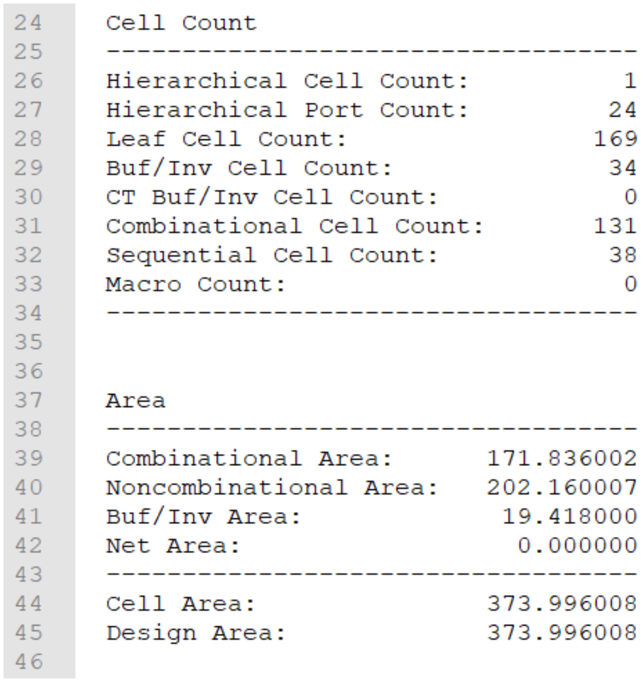


```
                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000            0.0000            0.0000            0.0000  (   0.00%)
memory             0.0000            0.0000            0.0000            0.0000  (   0.00%)
black_box          0.0000            0.0000            0.0000            0.0000  (   0.00%)
clock_network      0.0000            0.0000            0.0000            0.0000  (   0.00%)
register        3.2318e+03           1.9562        2.8696e+06        6.1031e+03  (  64.34%)
sequential         0.0000            0.0000            0.0000            0.0000  (   0.00%)
combinational     25.9603           99.1986        3.2573e+06        3.3824e+03  (  35.66%)
--------------------------------------------------------------------------------------------------
Total           3.2578e+03 uW      101.1548 uW     6.1269e+06 nW     9.4856e+03 uW
1
```

Figure [138]: Viterbi Decoder Power Reports

4.1.16. Cyclic Redundance Check

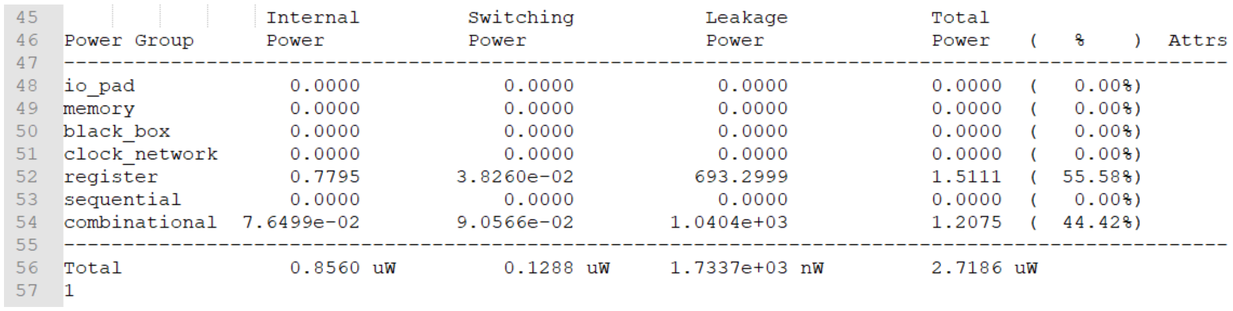
4.1.16.1. Area Results



```
Cell Count
-----------------------------------
Hierarchical Cell Count:          1
Hierarchical Port Count:         24
Leaf Cell Count:                169
Buf/Inv Cell Count:              34
CT Buf/Inv Cell Count:            0
Combinational Cell Count:       131
Sequential Cell Count:           38
Macro Count:                      0
-----------------------------------


Area
-----------------------------------
Combinational Area:      171.836002
Noncombinational Area:   202.160007
Buf/Inv Area:             19.418000
Net Area:                  0.000000
-----------------------------------
Cell Area:               373.996008
Design Area:             373.996008

```

Figure [139]: CRC Area Report

4.1.16.2. Power Results

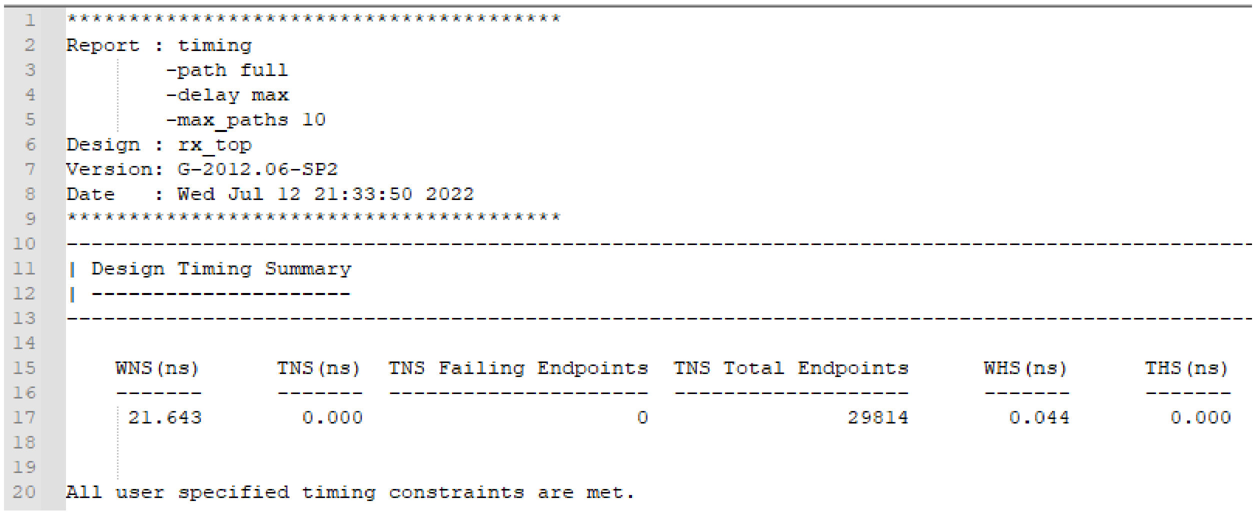


```
                 Internal          Switching           Leakage            Total
Power Group      Power             Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000            0.0000              0.0000             0.0000  (   0.00%)
memory             0.0000            0.0000              0.0000             0.0000  (   0.00%)
black_box          0.0000            0.0000              0.0000             0.0000  (   0.00%)
clock_network      0.0000            0.0000              0.0000             0.0000  (   0.00%)
register           0.7795        3.8260e-02            693.2999             1.5111  (  55.58%)
sequential         0.0000            0.0000              0.0000             0.0000  (   0.00%)
combinational  7.6499e-02        9.0566e-02          1.0404e+03             1.2075  (  44.42%)
--------------------------------------------------------------------------------------------------
Total              0.8560 uW         0.1288 uW       1.7337e+03 nW          2.7186 uW
1
```

Figure [140]: CRC Power Report

4.2. DC Results for RX Chain

4.2.1. Timing



```
****************************************
Report : timing
        -path full
        -delay max
        -max_paths 10
Design : rx_top
Version: G-2012.06-SP2
Date   : Wed Jul 12 21:33:50 2022
****************************************
------------------------------------------------------------------------------------------------
| Design Timing Summary
| ---------------------
------------------------------------------------------------------------------------------------

    WNS(ns)      TNS(ns)  TNS Failing Endpoints  TNS Total Endpoints      WHS(ns)      THS(ns)
    -------      -------  ---------------------  -------------------      -------      -------
     21.643        0.000                      0                29814        0.044        0.000


All user specified timing constraints are met.
```

Figure [141]: Timing Report

All specified timing constrains have been met with total available slack of **21.643 ns.**

4.2.2. Area Utilization

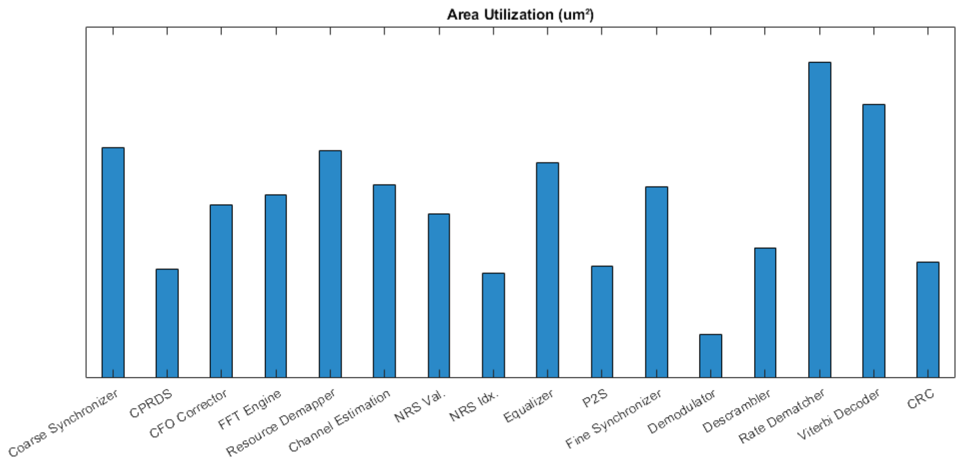


Figure [142]: Area Utilization Histogram

Table [26]: Area Utilization Table

Block	Area (μm²)
Coarse Synchronizer	135658.669893
CP Remover & Downsampler	270.256005
CFO Corrector	7014.952029
FFT Engine	11810.399929
Resource Demapper	112617.220342
Channel Estimation	19771.780108
NRS Value Generator	4502.582040
NRS Index Generator	215.726003
Channel Equalizer	61711.202107
Parallel to Serial	307.230001
Fine Synchronizer	18249.994066
Demodulator	9.576
Descrambler	798.798016
Rate Dematcher	1034782.81128
Viterbi Decoder	1226906.624348
CRC	373.996008

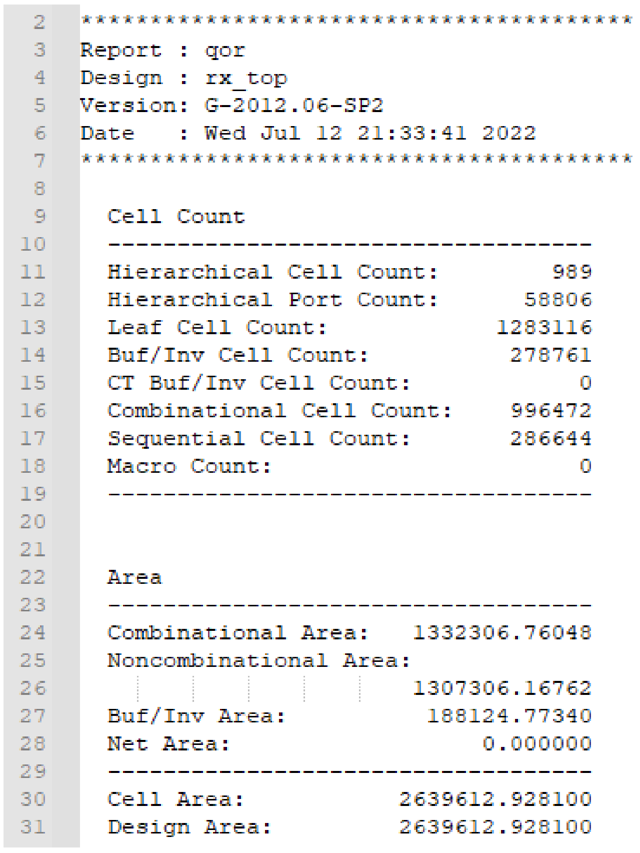

```
****************************************
Report : qor
Design : rx_top
Version: G-2012.06-SP2
Date   : Wed Jul 12 21:33:41 2022
****************************************


  Cell Count
  -----------------------------------
  Hierarchical Cell Count:        989
  Hierarchical Port Count:      58806
  Leaf Cell Count:            1283116
  Buf/Inv Cell Count:          278761
  CT Buf/Inv Cell Count:            0
  Combinational Cell Count:    996472
  Sequential Cell Count:       286644
  Macro Count:                      0
  -----------------------------------


  Area
  -----------------------------------
  Combinational Area:   1332306.76048
  Noncombinational Area:
                        1307306.16762
  Buf/Inv Area:          188124.77340
  Net Area:                  0.000000
  -----------------------------------
  Cell Area:           2639612.928100
  Design Area:         2639612.928100
```

Figure [143]: Area Utilization DC Report

Total Area Reported = 2639612.9281 μm²

4.2.3. Power Consumption

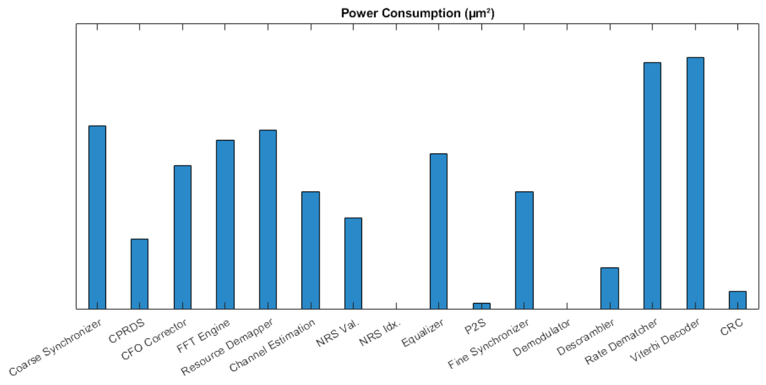
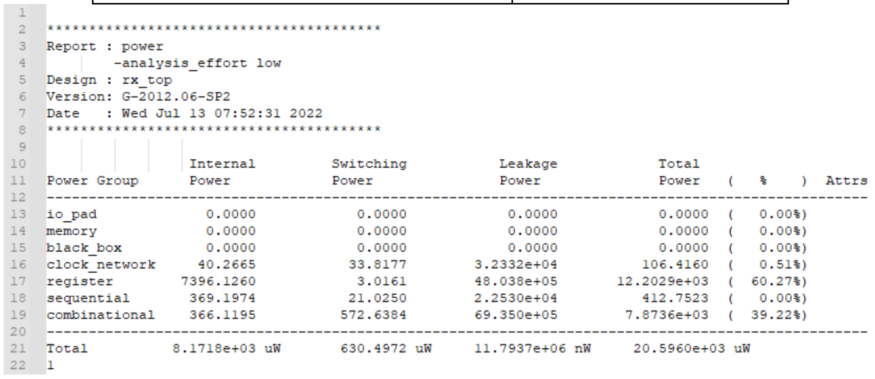


Figure [144]: Power Consumption Histogram

Table [27]: Total Power Consumption Table

Block	Total Power (µW)
Coarse Synchronizer	875.9543
CP Remover & Downsampler	16.9145
CFO Corrector	219.0724
FFT Engine	519.9951
Resource Demapper	752.9954
Channel Estimation	86.9436
NRS Value Generator	34.96
NRS Index Generator	1.21
Channel Equalizer	327.2816
Parallel to Serial	1.775
Fine Synchronizer	88.3618
Demodulator	1.05
Descrambler	6.0481
Rate Dematcher	7994.9
Viterbi Decoder	9485.6
CRC	2.7186



```

****************************************
Report : power
        -analysis_effort low
Design : rx_top
Version: G-2012.06-SP2
Date   : Wed Jul 13 07:52:31 2022
****************************************

                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000           0.0000             0.0000             0.0000  (   0.00%)
memory             0.0000           0.0000             0.0000             0.0000  (   0.00%)
black_box          0.0000           0.0000             0.0000             0.0000  (   0.00%)
clock_network     40.2665          33.8177         3.2332e+04           106.4160  (   0.51%)
register        7396.1260           3.0161         48.038e+05        12.2029e+03  (  60.27%)
sequential       369.1974          21.0250         2.2530e+04           412.7523  (   0.00%)
combinational    366.1195         572.6384         69.350e+05         7.8736e+03  (  39.22%)
--------------------------------------------------------------------------------------------------
Total          8.1718e+03 uW     630.4972 uW     11.7937e+06 nW     20.5960e+03 uW
1
```

Figure [145]: Total Power DC Report

Total Power Consumption = 20596 µW

4.3. FPGA Implementation Results

Targeted FPGA: **ZYNQ-7 ZC702 Evaluation Board**

4.3.1. Post-Implementation FPGA View

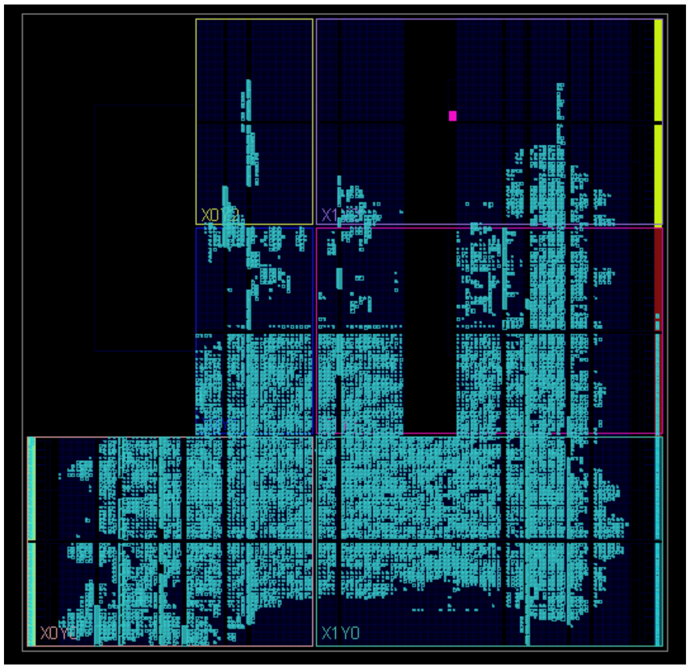
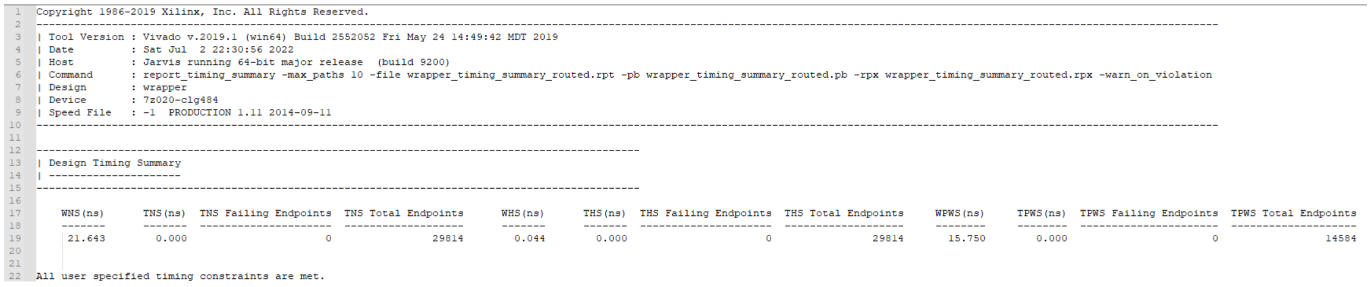


Figure [146]: Post-Implementation FPGA View

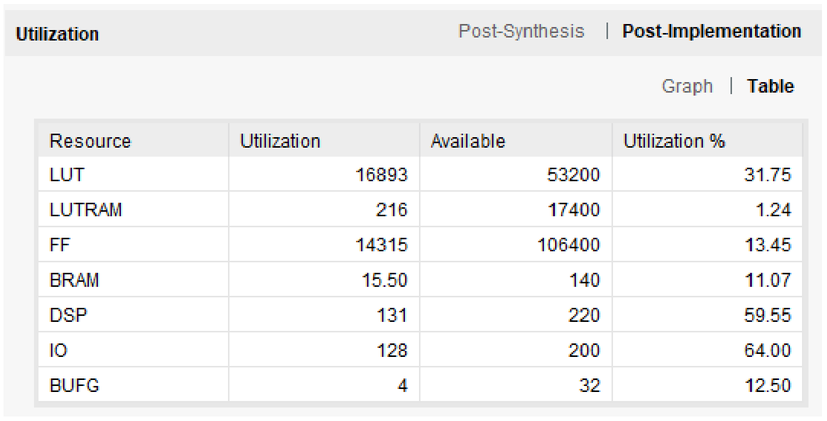
4.3.2. Timing



```
Copyright 1986-2019 Xilinx, Inc. All Rights Reserved.
-------------------------------------------------------------------------------------------------------------------------------------------------------------------
| Tool Version : Vivado v.2019.1 (win64) Build 2552052 Fri May 24 14:49:42 MDT 2019
| Date         : Sat Jul  2 22:30:56 2022
| Host         : Jarvis running 64-bit major release  (build 9200)
| Command      : report_timing_summary -max_paths 10 -file wrapper_timing_summary_routed.rpt -pb wrapper_timing_summary_routed.pb -rpx wrapper_timing_summary_routed.rpx -warn_on_violation
| Design       : wrapper
| Device       : 7z020-clg484
| Speed File   : -1  PRODUCTION 1.11 2014-09-11
-------------------------------------------------------------------------------------------------------------------------------------------------------------------


-----------------------------------------------------------------------------------
| Design Timing Summary
| ---------------------
-----------------------------------------------------------------------------------

    WNS(ns)      TNS(ns)  TNS Failing Endpoints  TNS Total Endpoints      WHS(ns)      THS(ns)  THS Failing Endpoints  THS Total Endpoints     WPWS(ns)     TPWS(ns)  TPWS Failing Endpoints  TPWS Total Endpoints
    -------      -------  ---------------------  -------------------      -------      -------  ---------------------  -------------------     --------     --------  ----------------------  --------------------
     21.643        0.000                      0                29814        0.044        0.000                      0                29814       15.750        0.000                       0                 14584


All user specified timing constraints are met.
```

Figure [147]: Vivado Timing Report

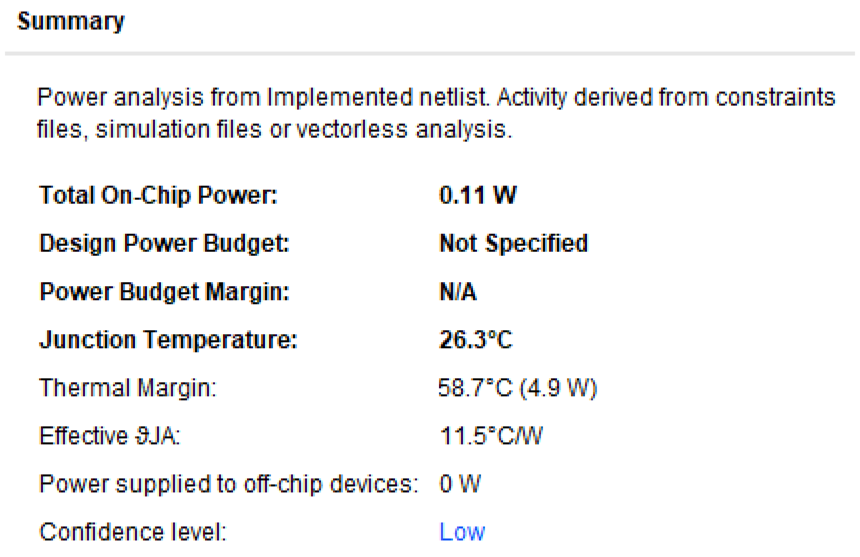
4.3.3. Area Utilization


Utilization
Post-Synthesis | Post-Implementation
Graph | Table

Resource	Utilization	Available	Utilization %
LUT	16893	53200	31.75
LUTRAM	216	17400	1.24
FF	14315	106400	13.45
BRAM	15.50	140	11.07
DSP	131	220	59.55
IO	128	200	64.00
BUFG	4	32	12.50

Figure [148]: FPGA Area Utilization

4.3.4. Power Consumption:



Summary

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**0.11 W**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**26.3°C**
Thermal Margin:	58.7°C (4.9 W)
Effective ϑJA:	11.5°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Figure [149]: FPGA Power Consumption Summary

5.1. FPGA Deployment

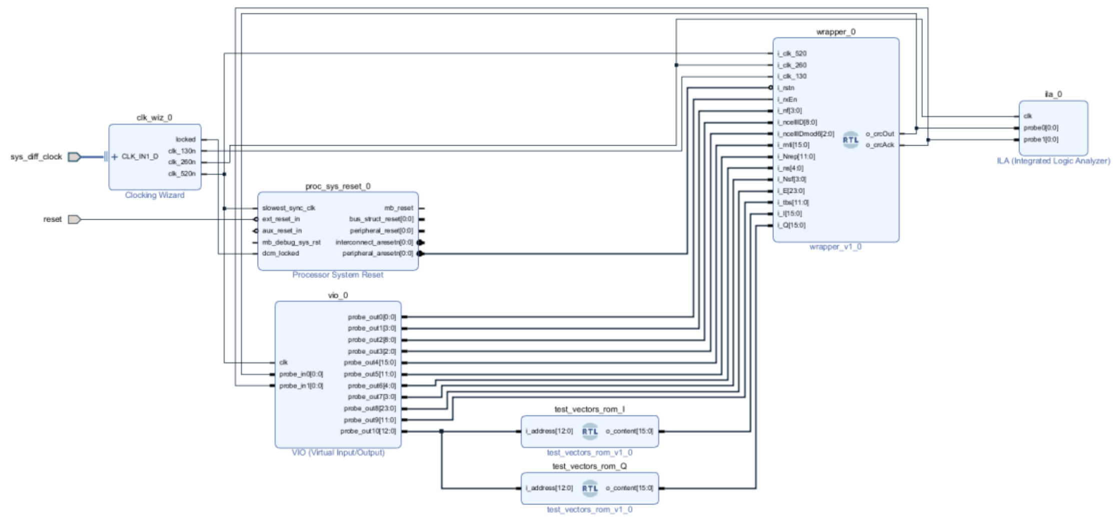


Figure [150]: FPGA Deployment Schematic

We are targeting **ZYNQ-7 ZC702 Evaluation Board**. This kit contains 3 user push buttons, 2 user switches and 8 user LEDs along with a reset switch. The available I/O's are not sufficient for our design to be deployed, so, we used built-in IPs in Vivado that allow us to use as many I/O pins as we need. These IPs are Virtual I/O (VIO), Integrated Logic Analyzer (ILA) and Clocking wizard and connected them with our design as shown in figure (149) above.

VIO: Virtual Input/Output (VIO) It is a configurable core IP that has real-time monitoring and driving capabilities for internal FPGA signals. To interact with the FPGA design, the input and output ports' size, number, and width may all be changed. The design outputs are the inputs into VIO, and vice versa.

ILA: Integrated Logic Analyzer (ILA) IP core is a logic analyzer core that can be used for analyzing and monitoring the internal signals of a design. ILA is used to visualize the design's outputs in a waveform.

Clocking Wizard: The Clocking Wizard simplifies the process of configuring the clocking resources in Xilinx FPGAs. Its input is an external differential clock and its output is the operating frequency for the design.

We assumed perfect synchronization while testing on the chain using 3 NB-IoT subframes using number of repetitions (Nrep) = 2 with data transport block size (TBS) = 136. As shown in the example shown in the figure below, the CRC Ack's indicating a successfully received transport block bits.

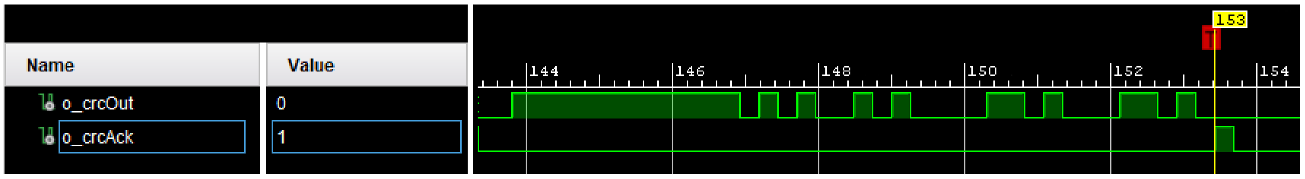


Figure [151]: RX Chain Output

We calculated receiver's performance and plotted the BER vs. SNR curve for the same Nrep and TBS as shown in the figure below.

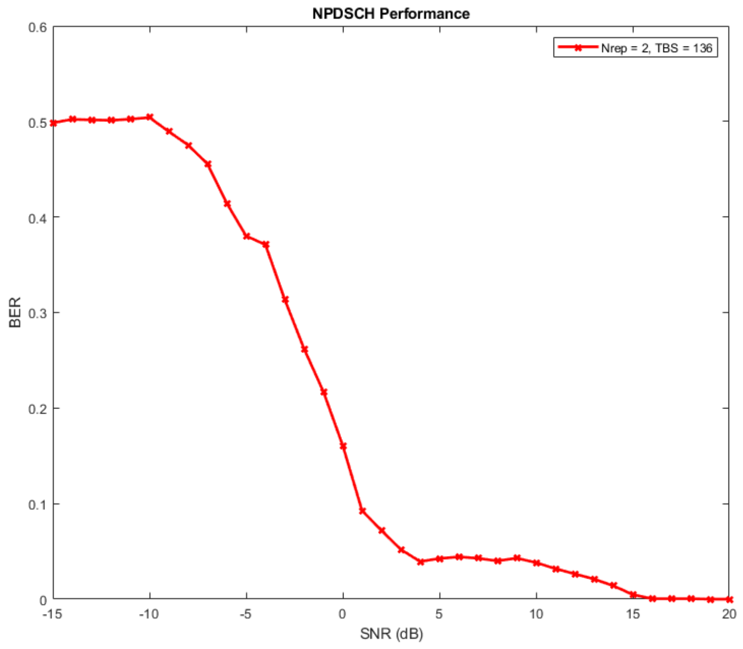


Figure [152152]: BER vs SNR Curve

References

[1] 3GPP TS 36.211 Evolved Universal Terrestrial Radio Access (E-UTRA); Physical channels and modulation (Release 14).

[2] 3GPP TS 36.212 Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and channel coding (Release 14).

[3] 3GPP TS 36.213 Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures (Release 14).

[4] Ali and W. Hamouda, "On the Cell Search and Initial Synchronization for NB-IoT LTE Systems," IEEE Comm. Lett., vol. 21, pp. 1843-1846, May 2017

[5] X. Wang, "Design and Implementation of CORDIC Algorithm Based on FPGA," 2018 International Conference on Robots & Intelligent System (ICRIS), 2018, pp. 70-71, doi: 10.1109/ICRIS.2018.00026.

[6] Andraka, Ray. (2001). A survey of CORDIC algorithms for FPGA based computers. ACM/SIGDA International Symposium on Field Programmable Gate Arrays - FPGA. 10.1145/275107.275139.

[7] Parallel Extensions to Single-Path Delay-Feedback FFT Architectures Brett W. Dickson, and Albert A. Conti.

[8] Y. E. Wang et al., "A Primer on 3GPP Narrowband Internet of Things (NB- IoT)," CoRR, vol. abs/1606.04171, 2016

[9] S. Adegbite, B. G. Stewart, and S. G. McMeekin, "Least Squares Interpolation Methods for LTE System Channel Estimation over Extended ITU Channels," *International Journal of Information and Electronics Engineering* vol. 3, no. 4, pp. 414-418, 2013.

[10] W. Liu and X. Li, "An Improved LMMSE Channel Estimation Algorithm of LTE System," 2012 Fourth International Conference on Computational and Information Sciences, 2012, pp. 231-234, doi: 10.1109/ICCIS.2012.71.

[11] Hala M. Abd Elkader, Gamal Mabrouk, Adly Tag* El-Dien and Reham S. Saad, Performance of LTE Channel Estimation Algorithms for Different Interpolation Methods and Modulation Schemes, 2014.

[12] A Tutorial on NB-IoT Physical Layer Design Matthieu Kanj, Vincent Savaux, Mathieu Le Guen, DOI 10.1109/COMST.2020.3022751, IEEE

[13] Magani, S., Kuchi, K. Cell-search and tracking of residual time and frequency offsets in low power NB-IoT devices. CSIT 7, 27–34 (2019).

[14] A Low-Power Implementation of arctangent function for Communication Applications using FPGA M. saber, Yutaka Jitsumatsu and T. Kohda.

[15] R. Y. Shao, Shu Lin and M. P. C. Fossorier, "Two decoding algorithms for tailbiting codes," in IEEE Transactions on Communications, vol. 51, no. 10, pp. 1658-1665, Oct. 2003, doi: 10.1109/TCOMM.2003.818084.

[16] J. Ortin, P. Garcia, F. Gutierrez and A. Valdovinos, "Simplified Circular Viterbi Algorithm for Tailbiting Convolutional Codes," 2011 IEEE Vehicular Technology Conference (VTC Fall), 2011, pp. 1-5, doi: 10.1109/VETECF.2011.6092864.