Cairo University
Faculty of Engineering
Electronics and Electrical
Communications Engineering
Department

ONE Lab

Si-VISION
INNOVATIVE IC SOLUTIONS

Bachelor Thesis

Narrowband IoT Physical Downlink Shared Channel Receiver

"NPDSCH RX R14"

Abanob Emil Sobhy

Abdallah Mahmoud Abostait

Ahmed Mohsen Abdel-Karim

Peter Emad Eskander

Aya Ahmed Magdy Ezz-Eldeen

Nouran Kassem Houssein Mohamed

Salma Ali Kamal

Under supervision of:

Associate prof. Mohamed Refki

Associate prof. Hassan Mostafa

Si-Vision

Acknowledgment

We wish to thank our supervisor: Dr. Hassan Mustafa, for his endless support in resources and encouragement. we are endlessly grateful to Si-Vision Inc. and ONE Lab, who supported, planned, and guided this project through all its phases, especially Eng. Ahmed Nasr El-Din, Eng. Ahmed Abdel-Motaal and Eng. Abdel-Rahman Hesham 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 Internet of Things IoT. This thesis provides a hardware implementation of the physical downlink shared channel PDSCH 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.

Table of content

Acknowledgment I
Abstract II
Table of content III
List of Figures: VIII
List of Tables: XIV
Abbreviation XV
Chapter 1 Introduction 1
1.1. NB-IoT applications and Usage 1
1.2. NPDSCH Receiver design based on Release 14 (3GPP): 2
1.3. Physical Resource Block Structure for NB-IoT 3
1.4. Frame Structure 4
1.5. The flow of data according to our NPDSCH chain design: 5
1.6. Thesis out Lines: 6
Chapter 2 Theory 7
2.1. Coarse Synchronizer [6:12, 53] 7
Problem Definition 7
Operating Conditions 7
NPSS Signal 7
Functionality outline 9
Algorithm 9
Constructing the Algorithm 11
2.2. CFO correction [13:15]: 12
CORDIC algorithm (coordinate rotation digital computer) 12
2.3. FFT [16:31]: 15
Algorithm: 15
2.4. RESOURCE DE-MAPPER [32]: 23
2.5. Channel Estimation [33:38]: 25
Channel Model for LTE and NB-IoT 25
Types of channel estimation 26
Channel Estimation Algorithms 28
2.6. NRS Value Generation: 30
2.7. NRS Location Generation: 31
2.8. Channel Equalizer: 32
Complex divider algorithm. 32
2.9. Fine Synchronization [39:40]: 33
2.9.1. Problem Definition: 33

Algorithm: 33
2.10. P/S and NRS removal: 34
2.11. Demodulation 35
2.12. Descrambling: 36
Descrambling Operation in NB-IoT 36
Descrambler initialization in NB-IoT: 36
2.13. Rate De-Matcher [41:50] 37
Rate Matcher 37
Rate De-Matcher: 39
2.14. Viterbi Decoder 41
Convolutional Codes 41
Tail Biting Convolutional Coding 42
Viterbi Decoder Description 43
1.14.4. Decoding of Convolutional Codes 46
2.15. Cyclic Redundancy Check 48
Cyclic Redundancy Check Algorithm 48
2.16. RAM Sharing 48
2.17. Design Specifications: 49
Chapter 3 Design implementation 50
3.1. Coarse Synchronizer 50
3.1.1. Block Diagram 50
3.1.2. State Diagram 51
3.1.3. Interface Tables 52
3.1.4. Results 54
3.2. CFO correction: 61
3.2.1. Block Diagram: 61
3.2.2. Interface Table: 61
3.2.3. Function of the design: 62
3.2.4. Design specification: 62
3.2.5. Detailed block diagram: 62
3.2.6. Design Interface: 63
3.2.7. Simulation Results: 63
3.3. FFT 66
3.3.1. Block Diagram: 66
3.3.2. Interface Table: 66

3.3.3. Function of the design: 67
3.3.4. Detailed implementation: 69
3.3.5. Design Interface: 70
3.3.6. Design Specification 71
3.3.7. Simulation Results: 71
3.4. RESOURCE DE-MAPPER 75
3.4.1. Block Diagram: 75
3.4.2. Interface Table: 75
3.4.3. Function of the design: 78
3.4.4. Design Interface: 80
3.4.5. Design Specification 80
3.4.6. Simulation Results: 81
3.5. Channel Estimation 85
3.5.1. Block Diagram: 85
3.5.2. Interface Table: 86
3.5.3. Function of the design: 87
3.5.4. Block Specification: 87
3.5.5. Detailed Design Implementation: 88
3.5.6. Design Interface: 90
3.5.7. Simulation Results: 90
3.6. NRS Value Generation 93
3.6.1. Block Diagram: 93
3.6.2. Interface Table: 93
3.6.3. Function of the design: 94
3.6.4. Block Specification: 94
3.6.5. Detailed Design Implementation: 95
3.6.6. Design Interface: 96
3.6.7. Simulation Results: 97
3.7. NRS Index Generation 99
3.7.1. Block Diagram: 99
3.7.2. Interface Table: 99
3.7.3. Function of the design: 100
3.7.4. Block Specification: 100
3.7.5. Detailed Design Implementation: 100
3.7.6. Design Interface: 102

3.7.7. Simulation Results: 102
3.8. Channel Equalizer 105
3.8.1. Block Diagram: 105
3.8.2. Interface Table: 105
3.8.3. Function of the design: 106
3.8.4. Block specification: 106
3.8.5. Detailed block diagram 106
3.8.6. Design Interface: 107
3.8.7. Simulation Results: 107
3.9. Fine Synchronization: 111
3.9.1. Block Diagram: 111
3.9.2. Interface Table: 111
3.9.3. Function of the design: 112
3.9.4. Block Specification: 114
3.9.5. Design Interface: 114
3.9.6. Simulation Results: 114
3.10. P/S and NRS removal: 117
3.10.1. Block Diagram: 117
3.10.2. Interface Table: 118
3.10.3. Function of the design: 118
3.10.4. Design specification: 118
3.10.5. Design Interface: 118
3.10.6. Simulation Results: 118
3.11. De-Modulation: 121
3.11.1. Block Diagram: 121
3.11.2. Interface Table: 121
3.11.3. Function of the design 122
3.11.4. Design Interface 122
3.11.5. Simulation Results: 122
3.12. Descrambling: 125
3.12.1. Block Diagram: 125
3.12.2. Interface Table: 125
3.12.3. Function of the design: 126
3.12.4. Design Interface: 126
3.12.5. Simulation Results: 126

3.13. Rate De-Matcher 130
3.13.1. Block Diagram 130
3.13.2. Interface Table 132
3.13.3. Function of the design [49] [50] [51] 132
3.13.4. Design Interface 133
3.13.5. Simulation Results 133
3.13.6. Synthesis Reports 137
3.14. Viterbi Decoder 138
3.14.1 Block Diagram: 138
3.14.2 Interface Table: 138
3.14.3 Function of the design: 139
3.14.4 Design Interface: 140
3.14.5 Simulation Results: 140
3.15. Cyclic Redundancy Check 146
3.15.1. Block Diagram 146
3.15.2. Interface Table: 147
3.15.3. Function of the design: 147
3.15.4. Design Interface: This block is communicating with: 147
3.15.5. Simulation Results: 147
3.16. RAM Sharing 150
3.16.1. Block Diagram: 150
3.16.2. Interface Table: 151
Chapter 4 System Integration and Results 153
4.1. MATLAB integration: 153
4.2. RTL integration: 155
4.3. Synthesis: 156
4.3.1. DC results: 156
4.3.2. VIVADO results: 159
4.4. FPGA implementation: 160
Chapter 5 Conclusion and Future Work 163
References 164

List of Figures:

Figure 1 Narrowband IoT usage **1**
Figure 2 NB-IoT deployment modes **2**
Figure 3 Range vs power consumption for LPWAN compared to other technologies **2**
Figure 4 Frame structure for NB-IoT LTE transceiver **3**
Figure 5 Radio frame structure type one **4**
Figure 6 NPDSCH RX Chain Block Diagram **5**
Figure 7 Mapping of Length-11 Code Cover to OFDM Symbols in Time-Domain **8**
Figure 8 Mapping of Base Sequence in Frequency-Domain (occupying 11 consecutive tones) **8**
Figure 9 Waveform of NPSS Symbols Generated at 1.92 MHz by Size-128 IFFT **9**
Figure 10 Algorithm High-Level State Diagram **10**
Figure 11 diagram show rotation of vector **13**
Figure 12 Radix-2 butterfly unit [25] **17**
Figure 13 Flow graph of 16-point Radix-2 FFT algorithm **17**
Figure 14 Radix-4 butterfly unit [26] **18**
Figure 15 Flow graph of 16-point Radix-4 FFT algorithm [26] **19**
Figure 16 Split-radix butterfly unit [29] **20**
Figure 17 Flow graph of 16-point SPLIT Radix FFT algorithm [30] **20**
Figure 18 Radix-2^2 butterfly unit [25] **21**
Figure 19 Flow graph of 16-point Radix-2^2 FFT algorithm [20] **21**
Figure 20 Resource block [4] **23**
Figure 21 LTE Frame Structure [4] **24**
Figure 22 Downlink resource grid **24**
Figure 23 Multi-path fading channel model **25**
Figure 24 Delay profile **25**
Figure 25 Delay profile for ETU according to 3GPP **26**
Figure 26 NB-IoT frame transmitted in an in-band mode over an LTE frame **27**
Figure 27 Channel effect on transmitted signal **28**
Figure 28 Zero-order interpolation **28**
Figure 29 Slot interpolation **29**
Figure 30 Sub-Frame interpolation **29**
Figure 31 BER vs SNR Matlab simulation for the three designs to get a rough estimation about the best performance **29**
Figure 32 R0: refer to the resource element placed at $NIDcell = 0$ **31**
Figure 33 Example for NRS location in subframe **33**
Figure 34 QPSK constellation **35**
Figure 35 Descrambling Algorithm **36**
Figure 36 Rate Matcher in transmitter **37**
Figure 37: Rate 1/3 tail biting convolutional encoder **41**
Figure 38: Convolutional Encoding with tail bits **42**
Figure 39: Trellis Transitions for Constraint length=4 ..... **43**
Figure 40: Error computation for constraints length=3 ..... **44**
Figure 41: State Diagram for constraints length=4. ..... **45**
Figure 42: Computed Distances for different paths **45**

Figure 43: Survivor path computation **46**
Figure 44: Trellis Transitions **47**
Figure 45: Trellis Transition at the end of the first iteration. **47**
Figure 46: Trellis at the end of the second iteration. **48**
Figure 47 CSYNCH Top-Level Block Diagram **50**
Figure 48 Simplified State Diagram of the Block **51**
Figure 49 CDF of PSS-Detection Latency in In-Band or Guard-Band Deployment **55**
Figure 50 PMF of Synchronization Timing Error in In-Band or Guard-Band Deployment **55**
Figure 51 CDF of Frequency Estimation Error in In-Band or Guard-Band Deployment **56**
Figure 52 CDF of PSS-Detection Latency in Standalone Deployment **56**
Figure 53 PMF of Synchronization Timing Error in Standalone Deployment **57**
Figure 54 CDF of Frequency Estimation Error in Standalone Deployment **57**
Figure 55 MATLAB Simulation Screenshot for Run 1 for CSYNCH **58**
Figure 56 RTL Simulation Screenshot for Run 1 for CSYNCH **58**
Figure 57 MATLAB Simulation Screenshot for Run 2 for CSYNCH **59**
Figure 58 RTL Simulation Screenshot for Run 2 for CSYNCH **59**
Figure 59 RTL Simulation Screenshot for Normal Operation for CSYNCH **59**
Figure 60 Screenshot from Synthesis QOR Report for CSYNCH **60**
Figure 61 Screenshot from Synthesis Power Report for CSYNCH **60**
Figure 62 Screenshot from Synthesis Area Report for CSYNCH **60**
Figure 63 CFO Block Diagram **61**
Figure 64 CFO Detailed Block Diagram **62**
Figure 65 : CORDIC block diagram **62**
Figure 66 recursive pipelined CORDIC design **63**
Figure 67 MATLAB result for different input offset CFO **63**
Figure 68 RTL simulation results of CFO **64**
Figure 69: comparing results of RTL and MATLAB in CFO **64**
Figure 70: average error between RTL and MATLAB in CFO **64**
Figure 71: Area report of CFO **65**
Figure 72 : Power report of CFO **65**
Figure 73: Timing report of CFO **66**
Figure 74 16-point FFT radix22 SDF block diagram **66**
Figure 75 16-point FFT Block diagram before optimized **67**
Figure 76 Optimized 16-point FFT block diagram **68**
Figure 77 Stage 1&4 after optimization block diagram of FFT **69**
Figure 78 Stage 2 block diagram of FFT **69**
Figure 79 Stage 3 block diagram of FFT **70**
Figure 80 interfacing blocks of FFT **70**
Figure 81 16-point FFT MATLAB output of FFT **71**
Figure 82 12 FFT output stored in resource de-mapper of FFT **71**
Figure 83 16-point FFT RTL simulation of FFT **71**
Figure 84 FIFO RTL simulation of FFT **72**
Figure 85 Power Report of memory before FFT **72**
Figure 86 Power Report of FFT **73**
Figure 87 Area Report of memory before FFT **73**
Figure 88 Area Report of FFT **74**
Figure 89 Timing Report of memory before FFT **74**

Figure 90 Timing Report of FFT **74**
Figure 91 resource element de-mapper block diagram **75**
Figure 92 Time multiplexing between NB-IoT downlink physical channels and signals [3] **79**
Figure 93 interface block diagram **80**
Figure 94 resource element de-mapper MATLAB results **81**
Figure 95 resource element de-mapper RTL simulation **81**
Figure 96 Storage element memory **82**
Figure 97 Resource De-mapper memory **82**
Figure 98 Storage Element Power Report **82**
Figure 99 Resource element De-mapper Power Report **83**
Figure 100 Storage Element Area Report **83**
Figure 101 Resource Element De-Mapper Area Report **84**
Figure 102 Storage Element Timing Report **84**
Figure 103 Resource Element De-Mapper Timing Report **84**
Figure 104 Channel Estimation Block Diagram **85**
Figure 105 Channel Estimation Block diagram **87**
Figure 106 Channel Estimation-Get Data Block Diagram **88**
Figure 107 Channel Estimation-Complex Multiplier Block Diagram **88**
Figure 108 Channel Estimation-Register file Block Diagram **89**
Figure 109 Channel Estimation-Interpolation Block Diagram **89**
Figure 110 Channel Estimation interface block diagram **90**
Figure 111 Channel Estimation performance **90**
Figure 112 RTL Channel estimation real outputs with fixed-point approximation **91**
Figure 113 RTL Channel estimation area report **91**
Figure 114 RTL Channel estimation power report **92**
Figure 115 RTL Channel estimation timing report **92**
Figure 116 NRS Generator Block Diagram **93**
Figure 117 NRS values generator Block diagram **94**
Figure 118 NRS generator Second m-sequence Generator block diagram **95**
Figure 119 NRS Generator LFSR Block diagram **95**
Figure 120 NRS generator Register files block diagram **96**
Figure 121 NRS values generator interface Block diagram **96**
Figure 122 NRS Values generated by built in MATLAB function for sub frame number = 6 ..... **97**
Figure 123 NRS Values generated by created MATLAB function for sub frame number = 6 ..... **97**
Figure 124 NRS Values generated by RTL block **97**
Figure 125 NRS Values generated from the MATLAB **97**
Figure 126 NRS Values generation block power report **98**
Figure 127 NRS Values generation block area report **98**
Figure 128 NRS Values generation block timing report **98**
Figure 129 NRS Index Generator Block Diagram **99**
Figure 130 NRS Location generator Block diagram **100**
Figure 131 NRS index GEN mod-6 Block Diagram **100**
Figure 132 NRS index GEN indices Generator **101**
Figure 133 NRS Locations Generator **101**
Figure 134 NRS Location generator Block diagram **102**
Figure 135 figure of the MATLAB function output for $NIDcell = 1$ **102**
Figure 136 figure of the MODELSIM function output for $NIDcell = 5$ **103**

Figure 137 NRS Location Generator block power report **103**
Figure 138 NRS Location Generator block area report **103**
Figure 139 NRS Location Generator block timing report **104**
Figure 140 : Channel equalizer block diagram **105**
Figure 141 : full detailed diagram of equalizer **106**
Figure 142: comparison between input symbols and the output after equalization **107**
Figure 143: inputs and outputs of equalizer **107**
Figure 144: output of RTL of channel Equalizer **108**
Figure 145: comparison between output from MATLAB and RTL **108**
Figure 146: average error between MATLAB and RTL results **109**
Figure 147: area report of channel equalizer **109**
Figure 148: timing report of channel equalizer **109**
Figure 149: power report of channel equalizer **110**
Figure 150 Fine Synchronization Block Diagram **111**
Figure 151 Fine Synchronization Detailed Block Diagram **112**
Figure 152 Arctan Block Diagram **112**
Figure 153 Arctan Curve **113**
Figure 154 NRS in subframe **114**
Figure 155 Values of NRS **114**
Figure 156 Fine Synchronization MATLAB Output **115**
Figure 157 Fine Synchronization RTL Output **115**
Figure 158 Fine Synchronization Area Report **116**
Figure 159 Fine Synchronization Power Report **116**
Figure 160 Fine Synchronization Timing Report **117**
Figure 161: output of P/S and NRS removal block **118**
Figure 162: output of P/S and NRS removal **119**
Figure 163: area report of P/S and NRS removal block **119**
Figure 164: timing report of P/S and NRS removal block **119**
Figure 165: power report of P/S and NRS removal block **120**
Figure 166 Demapper Block Diagram **121**
Figure 167 Input to demapper from p/s & NRS removal **122**
Figure 168 MATLAB Output from Demapper **122**
Figure 169 RTL Output from Demapper **123**
Figure 170 Demapper Area Report **123**
Figure 171 Demapper Power Report **124**
Figure 172 Demapper Timing Report **124**
Figure 173 Descrambling Block Diagram **125**
Figure 174 Input to Descrambling from De-mapper **126**
Figure 175 MATLAB Output from Descrambling for NcellID = 0, Ns = 2, Nf = 100 ..... **127**
Figure 176 MATLAB Output from Descrambling for NcellID = 0, Ns = 3, Nf = 100 ..... **127**
Figure 177 MATLAB Output from Descrambling for NcellID = 0, Ns = 3, Nf = 101 ..... **127**
Figure 178 MATLAB Output from Descrambling for NcellID = 14, Ns = 3, Nf = 101 ..... **128**
Figure 179 RTL Output from Descrambling for NcellID = 0, Ns = 2, Nf = 100 ..... **128**
Figure 180 RTL Output from Descrambling for NcellID = 0, Ns = 3, Nf = 100 ..... **128**
Figure 181 RTL Output from Descrambling for NcellID = 0, Ns = 3, Nf = 101 ..... **128**
Figure 182 RTL Output from Descrambling for NcellID = 14, Ns = 3, Nf = 101 ..... **128**
Figure 183 Descrambling Area Report **129**

Figure 184 Descrambling Power Report **129**
Figure 185 Descrambling Timing Report **130**
Figure 186: Rate De-Matcher Block Diagram **130**
Figure 187: Bit Collection Block Diagram **131**
Figure 188 De-interleaver Block Diagram **131**
Figure 189Rate De-matcher Output data from MATLAB block for E = 60 ..... **133**
Figure 190:Rate De-matcher Output data from MATLAB function for E = 60 ..... **134**
Figure 191:Rate De-matcher Output data from MATLAB block for E = 120 ..... **134**
Figure 192 Rate De-matcher Output data from MATLAB function for E = 120 ..... **135**
Figure 193 Rate De-matcher Output data from MATLAB block for E = 240 ..... **135**
Figure 194 Rate De-matcher Output data from MATLAB function for E = 240 ..... **135**
Figure 195 Rate De-matcher RTL output for E = 120 ..... **136**
Figure 196 Rate De-matcher MATLAB dataout1 ..... **136**
Figure 197 Rate De-matcher MATLAB dataout2 ..... **136**
Figure 198 Rate De-matcher MATLAB dataout3 ..... **137**
Figure 199 Rate De-matcher Synthesis area report **137**
Figure 200 Rate De-matcher Synthesis power report **137**
Figure 201 Rate De-matcher Synthesis Timing report **137**
Figure 202: Trellis Structure **140**
Figure 203: Next states in Trellis Diagram for (Rate=1/3, K=7) **141**
Figure 204: Trellis Diagram Expected Outputs **142**
Figure 205: Path metric Results **143**
Figure 206: Trace back start point **143**
Figure 207: Hamming Distances **143**
Figure 208: Adder, Compare and select unit results. **144**
Figure 209: Decoded sequence **144**
Figure 210: Power Report **145**
Figure 211: Area Report **145**
Figure 212: Timing Report **145**
Figure 213 CRC Block Diagram **146**
Figure 214: CRC Detailed Block Diagram **146**
Figure 215: output CRC block **147**
Figure 216: output CRC function **148**
Figure 217: output CRC RTL **148**
Figure 218: output CRC function MATLAB **148**
Figure 219: CRC Synthesis power report **149**
Figure 220: CRC Synthesis timing report **149**
Figure 221: CRC Synthesis area report **149**
Figure 222 Shared RAM Block Diagram **150**
Figure 223 MATLAB BER Vs SNR **153**
Figure 224 MATLAB BLER Vs SNR **154**
Figure 225 BLER Vs SNR for MATLAB and RTL designed model compared with ideal model **154**
Figure 226 RTL BER Vs SNR **155**
Figure 227 RTL BLER Vs SNR **155**
Figure 228: Area report for each block **156**
Figure 229: Power report for each block **156**
Figure 230: Latency for each block **157**

Figure 231 Whole Chain DC Area Report **157**
Figure 232 Whole Chain DC Power Report **158**
Figure 233 Whole Chain DC Timing Report **158**
Figure 234 VIVADO Utilization for Vertex 7 for Whole Chain **159**
Figure 235 VIVADO power summary for Vertex 7 for Whole Chain **159**
Figure 236 VIVADO Behavioral Simulation **160**
Figure 237 ILA FPGA Output **161**
Figure 238 VIO FPGA Output **161**
Figure 239 FPGA Output if reset is 0 ..... **162**
Figure 240 FPGA Output if enable 0 ..... **162**

List of Tables:

Table 1 Comparison of Pipelined FFT Architectures [17] **16**
Table 2 FFT different radixes comparison **23**
Table 3 QPSK constellation **35**
Table 4 Permutation Table **38**
Table 5 Design Specifications **49**
Table 6 Usage of Important Components and Resources: **50**
Table 7 Illustration of Transitions between States: **52**
Table 8 csynch Interfacing Table of the Top-Level Module: **52**
Table 9 Block's Interfacing Table with Shared-Ram: **52**
Table 10 csynch RTL Simulation Results Summary: **54**
Table 11 Summary of the Synthesis Results for csynch: **58**
Table 12 interface table of CFO correction **61**
Table 13 interface table of FFT **66**
Table 14 FFT comparison with [25] **75**
Table 15 Storage Element Interface Table **75**
Table 16 Resource De-mapper interface table **76**
Table 17 Bit-reversing operation **79**
Table 18 Channel Estimation interface table **86**
Table 19 Channel Estimation interface table **91**
Table 20 NRS generator interface table **93**
Table 21 NRS Index Generator interface table **99**
Table 22 MATLAB results for NRS index with $NIDcell = 1$ **102**
Table 23 MATLAB results for NRS index with $NIDcell = 5$ **103**
Table 24: interface table of channel equalizer **105**
Table 25 Fine Synch interface table **111**
Table 26 P/S and NRS removal interface table **118**
Table 27 De-modulator interface table **121**
Table 28 Descrambler Interface table **125**
Table 29 De-Rate Matcher Interface Table **132**
Table 30: Viterbi Decoder **138**
Table 31: CRC Interface Table **147**
Table 32 Shared RAM Interface Table **151**

Abbreviation

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

UE User Equipment

Z-F Zero-Forcing

Chapter 1
Introduction

1.1. NB-IoT applications and Usage

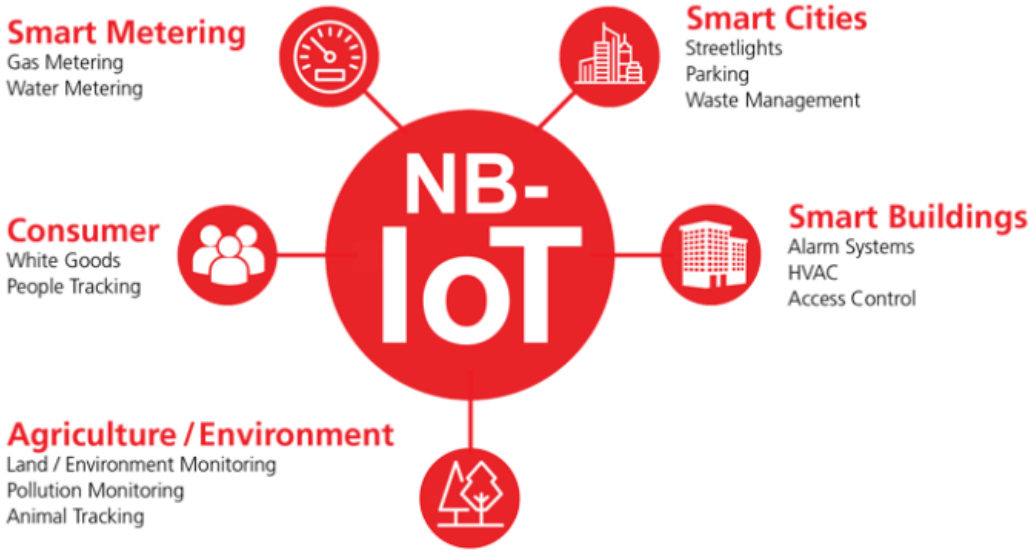


Figure 1 Narrowband IoT usage

Internet of Things is a network of physical objects or people called things, where electronics, software and network work together to open the door for these things to be able to collect and exchange data remotely. There is a lot of application for IoT devices such as smart cities, wearables and Health Monitoring, security and surveillance. There are some requirements that are needed in order to support the IoT devices for instance low cost and long battery life. Moreover, there are some requirements on the network like, low data and low latency support, and extended coverage link budget. 3GPP also has taken some evolutionary steps regards the network and devices in order to meet the connectivity of emerging to IoT segment. There are solutions to meet the IoT requirements such as LTE-M and NB-IoT. [1]

The narrow band IoT proposal is set for approval in 3GPP Rel14 with following improvements. NB-IoT or NB-LTE is a new 3GPP radio-access technology, which is designed to achieve the perfect co-existence performance along with GSM, GPRS, and LTE technologies. One of the advantages of using LTE is the mobility. It is also very popular as it has high data rate, high capacity and spectrum efficiency. For NB-LTE Rel. 14 occupies 180 KHz of the spectrum.

Modes of operation:

- In-band:

In this mode, 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.

- Guard band:

In this mode, an NB-IoT carrier is deployed within the guard band of an LTE carrier by using used Resource Blocks within Blocks within LTE carrier Guard Band and it does not take any capacity from the main LTE traffic carrier.

- Standalone:

In this mode, an NB-IoT carrier is deployed independently of any LTE carrier and it can work as a replacement to GSM carriers. It also can provide deployment flexibility based on available spectrum and use cases [2].

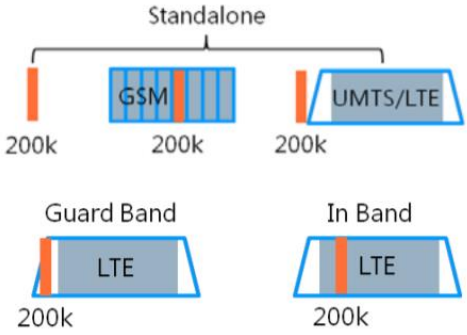


Figure 2 NB-IoT deployment modes

1.2. NPDSCH Receiver design based on Release 14 (3GPP):

The project aims to implement and, simulate the Narrowband Physical Downlink Shared Channel NPDSCH receiver based on release 14 for Third-Generation Partnership Project 3GPP.

NB-IoT is a cellular radio access technology based on Long Term Evolution LTE for Low-Power Wide-Area Network LPWAN enables low data rates power and cost.

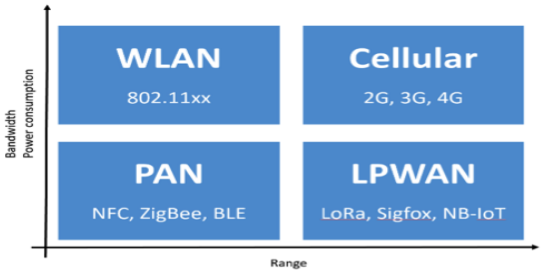


Figure 3 Range vs power consumption for LPWAN compared to other technologies

In 2014 through 2015, there are some proposals for NB until 2016 when 3GPP include all proposals as a work item for release 13, then in mid of 2016 NB-IoT was recognized in release 13 followed by updated versions with different Enhancements.

For quality of service improvement, release 14 come with more updates; New power class for the maximum output User Equipment reduced to 14 dB, New Transport Block size support reach 2536 bits to improve data rates and introduce the ability of second HARQ to enhance the reliability of the links for the UEs.

1.3. Physical Resource Block Structure for NB-IoT

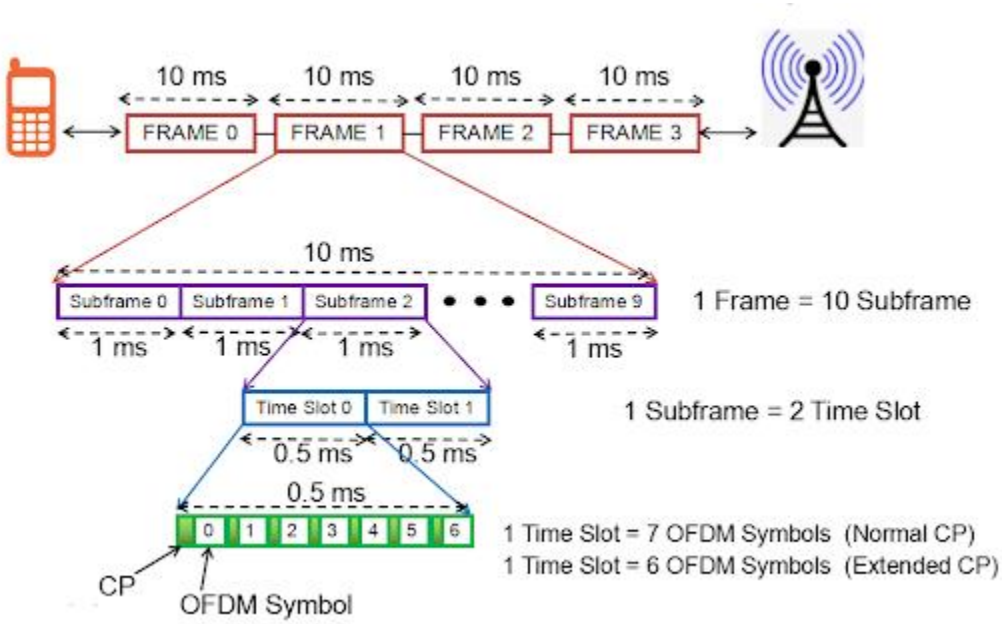


Figure 4 Frame structure for NB-IoT LTE transceiver

The transmitted signal in each slot is described by one or several resource grids of $N_{RB}^{DL} N_{sc}^{RB}$ subcarriers and N_{symb}^{DL} OFDM symbols. Where each resource grid consists of $N_{symb}^{DL} \times N_{sc}^{RB}$ resource elements. The quantity N_{RB}^{DL} depends on the downlink transmission bandwidth configured in the cell and shall fulfil

$$N_{\mathrm{RB}}^{\mathrm{min,DL}} \le N_{\mathrm{RB}}^{\mathrm{DL}} \le N_{\mathrm{RB}}^{\mathrm{max,DL}}$$

Where $N_{\mathrm{RB}}^{\mathrm{min,DL}} = 6$ and $N_{\mathrm{RB}}^{\mathrm{max,DL}} = 110$ are the smallest and largest downlink bandwidths, respectively, supported by the current version of this specification.

Resource block is also known as slot and its transmission time is about 0.5ms. Two slots are combining together forming subframe and ten subframes are combining together forming a whole frame and it takes 10ms to transmit the whole frame where each subframe needs 1ms to be transmitted. Moreover, the length of slot differs depends on the CP as in Normal CP slot contains 7 OFDM symbols and the time taken by the first symbol is greater than the rest as discussed in section 1.3 while in Extended CP the slot contains 6 OFDM symbols and the slot's time is divided equally between the symbols.

For NB-IoT Normal Cyclic Prefix is used and the number of Resource blocks used is 1 Resource Block for the NB-IoT with 12 frequencies sub-carriers and 14 symbols for each sub frame with total BW=180 KHz.

1.4. Frame Structure

According to 3GPP there are three radio frame structures designed for LTE:

The radio frame structure type 1:

Is only applicable to FDD (for both full duplex and half duplex operation) and has a duration of 10ms and consists of 20 slots with a slot duration of 0.5ms. Two adjacent slots form one sub-frame of length 1ms, except when the sub-carrier bandwidth is 1.25 kHz, in which case one slot forms one sub-frame.

The radio frame structure type 2:

Is only applicable to TDD and consists of two half-frames with a duration of 5ms each and containing each either 10 slots of length 0.5ms, or 8 slots of length 0.5ms and three special fields (DwPTS, GP and UpPTS) which have configurable individual lengths and a total length of 1ms.

The radio frame structure type 3:

Is only applicable to LAA secondary cell operation. It has a duration of 10ms and consists of 20 slots with a slot duration of 0.5ms. Two adjacent slots form one sub-frame of length 1ms. Any sub-frame may be available for downlink or uplink transmission.

For narrow band Downlink Physical shared channel, frame 1 is used.

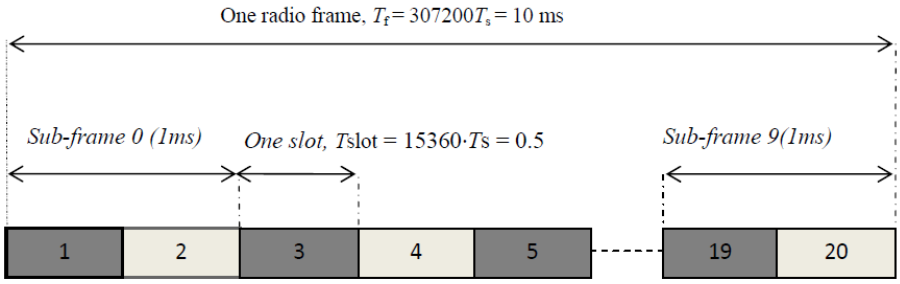


Figure 5 Radio frame structure type one

Form the shown figure the frame time $T_f = 307200\, T_s = 10ms$, $T_{subframe} = 30720\, T_s = 1ms$

According to 3GPP

$$T_s = {}^{1}/_{(15000)(2048)}\, ms$$

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

$$T_u = 66.67\ us, \qquad T_{symbol} = {}^{1ms}/_{14} = 71.4us$$

Number of samples for $cp = 160\ for\ 1st\ OFDM\ symbol\ and\ 144\ for\ the\ rest$

So the Minimum number of samples used in LTE is 128 to get integer number of samples for $cp = 10\ for\ the\ 1st\ OFDM\ symbol\ and\ 9\ for\ the\ rest$

For NB-IoT LTE it follows the numbers of the minimum samples which is 128 although its number of sub-carriers is 12 so we use a down sampler in our design from 128 samples to 16 samples.

Thus, the sampling rate calculations become $T_s = {}^{1}/_{(15000)(128)} = {}^{1}/_{1.92}\, ms$ for NB-IoT LTE.

1.5. The flow of data according to our NPDSCH chain design:

In our design for the chain of NPDSCH, we take into consideration the effect of these enhancements on the physical layer and finally reach for this Block Diagram for our chain:

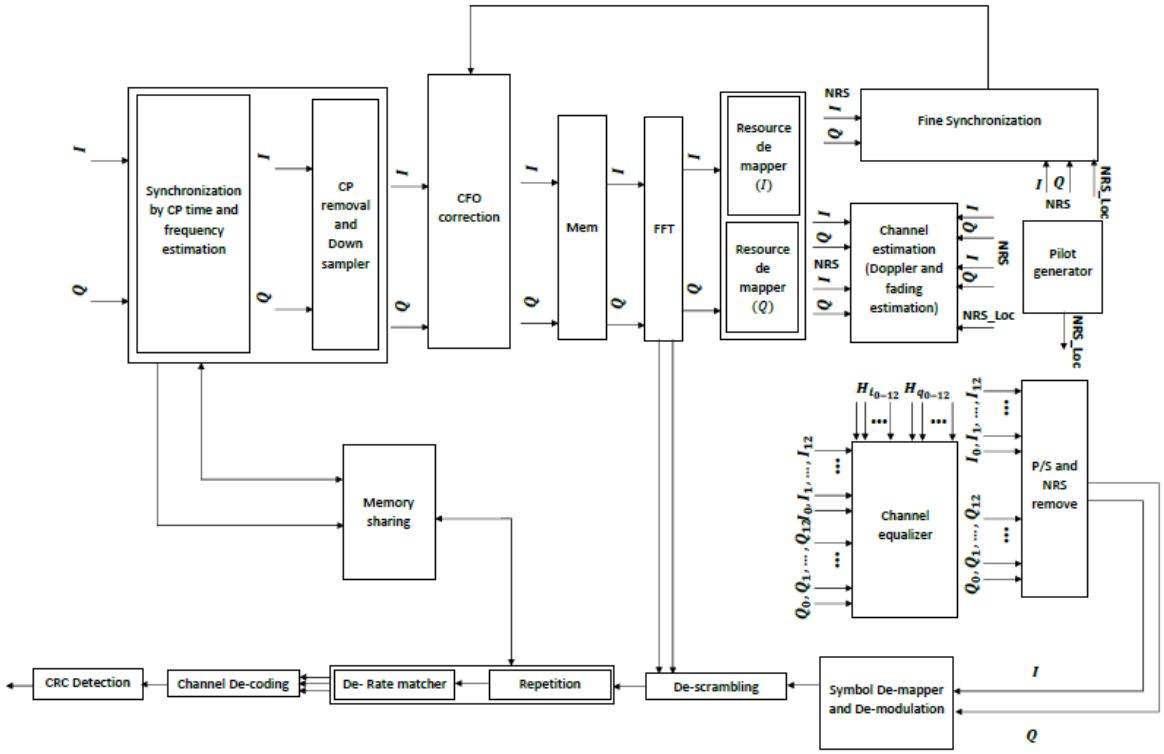


Figure 6 NPDSCH RX Chain Block Diagram

The in-phase and quadrature-phase data input waveform starts to enter the NPDSCH receiver device through the Coarse Synchronization Block, after the synchronization with the transmitter happen CP removed and data went through down sampler then finally to CFO correction block so the frequency offset estimation resulted from the Coarse Synchronization used in CFO correction.

FIFO based Memory used in storing the 16 OFDM symbols for preparation to the FFT process that transform the signal from time domain to frequency domain storing the output in two consecutive memories called Resource De-mapper.

Channel Estimation estimate the channel frequency response using the pilots stored in the Resource De-mapper and pass the output to the equalizer for compensate the channel effect.

Fine synchronization is used to keep tracking time and frequency offset.

Data then pass through NRS removal block to remove pilots and transform parallel data flow into serial data flow.

Real and Imaginary symbols pass through De-modulation to converts into bits based on QBSK constellation and go to the scrambler in order to randomize the stream of data.

Bit stream finally go through Rate De-matcher block to Improve channel efficiency by changing the code rate of the transmitted data, then pass through the channel De-coder to correct the received bit based on a hard decision decoder and, finally reach the CRC block to get the ACK signal to the upper layer.

1.6. Thesis out Lines:

This thesis is divides into five chapters:

Chapter 1: The introduction to NB-IoT LTE, Release 14 enhancements, our chain block diagram and, brief description for the functionality of each block.
Chapter 2: The Theory Background and the Algorithms used in implementing each block with detailed explanations for the function and the role of each block.
Chapter 3: The block diagrams for each block design, Matlab results compared to simulation results, and power and area synthesis results.
Chapter 4: The Matlab and RTL full Integration plain took to integrate the whole Matlab blocks' functions and the RTL blocks, also include the MATLAB, Simulation and, power and area Synthesis Integration Results for the whole chain and, FPGA Implementation and results.
Chapter 5: The conclusion of our design specs, improvements done in our design and, the future works.

Chapter 2
Theory

2.1. Coarse Synchronizer [6:12, 53]

Problem Definition

2.1.1.1. OFDM-Based Systems Sensitivity to Frequency and Time Offsets

Communication systems based on OFDM struggle with two main issues: ICI, inter-carrier interference and ISI, inter-symbol interference. Firstly, the high sensitivity to ICI is due to a basic assumption in OFDM systems, namely that signals on subcarriers are orthogonal to each other. That assumption means that the spectrum of an arbitrary signal on an arbitrary subcarrier must have nulls at all other subcarriers' frequencies. Hence, a frequency shift introduced to this system can undermine this assumption, and in this case, there is ICI, that implies that the signals on different subcarriers interfere with one another, which degrades the SINR. Secondly, the sensitivity to ISI is not unique to OFDM systems; it is a common issue in communications systems generally. ISI occurs when two symbols interfere in the time-domain as they, for example, are dispersed due to propagation through a multipath channel or due to poor pulse shaping at the transmitter, that leads to overlapping between symbols and hence degrades the system performance. However, as these previous problems are usually dealt with in other contexts, the ISI at hand is mainly caused by another issue, that is synchronization error, which simply means to mistake one time-domain sample as the beginning of a symbol when it is not.

2.1.1.2. Frequency and Time Offsets: Sources and Ranges

ICI and ISI have multiple sources. There are two main sources of ICI: CFO, which is the difference between the carrier frequency at the transmitter and the receiver RF oscillators and the other 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 an "anchor carrier", this carrier is searched for on a 100KHz raster, so the offset between the 100KHz raster and the center frequency of the anchor carrier is the raster offset. As for the ranges of frequency offset sources, the carrier frequency offset can take a value up to 18KHz assuming a maximum mismatch of 20ppm between the transmitter and receiver oscillators and a carrier frequency around 900MHz, on the other hand, 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. Hence, the maximum absolute frequency offset is assumed to be 25.5KHz [6] [7]. Considering ISI, for this context, perfect pulse shaping is assumed; also, it is assumed that the CP is longer than the multipath channel's maximum excess tap delay. The remaining source is the synchronization error: incorrectly defining the beginning of symbols. Initially, the information about the beginning of symbols is not available to the receiver and must be acquired through the synchronization process, thus errors in estimating the beginning of symbols in the synchronization process are the source of ISI of interest in this context.

Operating 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 [7]. The multipath fading channel is modeled by the ETU model having 9 taps with a maximum excess tap delay of 5us and a maximum Doppler frequency of 5Hz due to the stationary nature of the applications of NB-IoT [8].

NPSS Signal

The NPSS is a signal composed of concatenated short Zadoff-Chu sequences designed to facilitate the frequency and time offsets estimation. It is mapped to the last 11 symbols in the 6^{th} subframe in every radio frame and thus has a periodicity of 10ms, the mapping excludes the first 3 OFDM symbols of the subframe to avoid interference with legacy LTE for in-band deployment [7]. The signal is constructed in two layers: a base sequence across eleven consecutive subcarriers in the frequency-domain and a code cover across eleven OFDM symbols in the time-domain.

The base sequence $d_l(n)$ is defined as follows [9] [7],

$$d_l(n) = S(l) \cdot e^{-j\frac{\pi u n(n+1)}{11}}, \qquad u = 5, \qquad n = 0, 1, \dots, 10 \tag{1}$$

The code cover $S(l)$ is defined as follows [9] [7],

$$S(0{:}\,10) = \{1, 1, 1, 1, -1, -1, 1, 1, 1, -1, 1\}, \tag{2}$$

To illustrate the mapping process, **Error! Reference source not found.** shows the mapping of the code c over to time-domain OFDM symbols and Figure 8 shows the mapping of the base sequence to frequency-domain subcarriers [7].

LTE Symbol	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Code Cover $S(l)$				1	1	1	1	-1	-1	1	1	1	-1	1
NB-PSS Symbol				A	A	A	A	-A	-A	A	A	A	-A	A

Figure 7 Mapping of Length-11 Code Cover to OFDM Symbols in Time-Domain

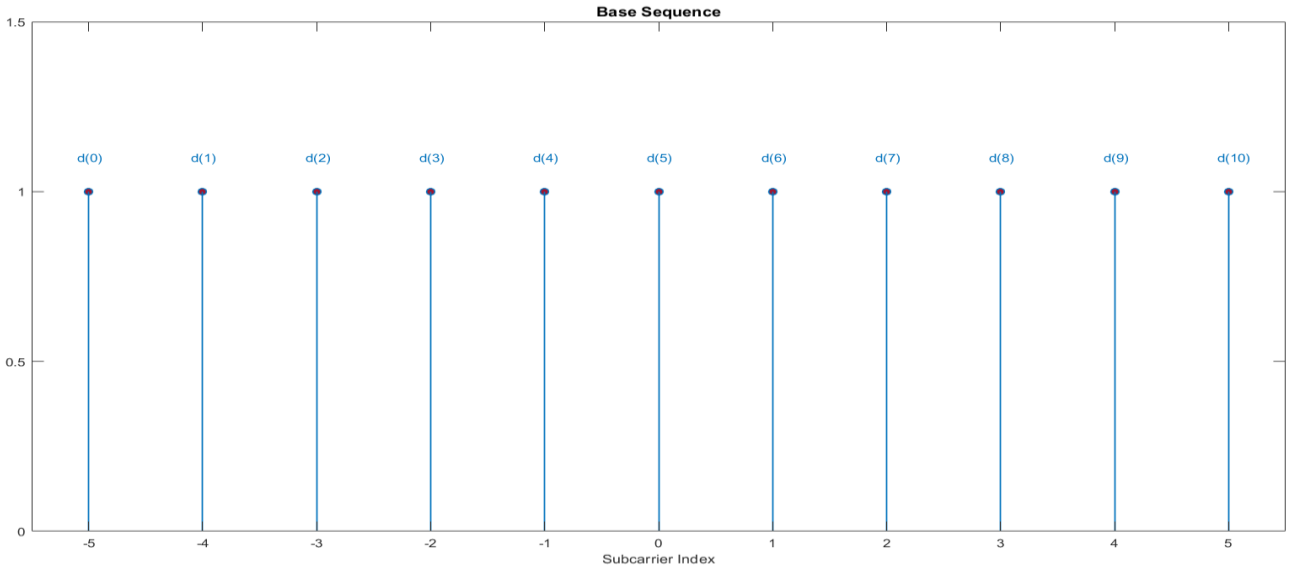


Figure 8 Mapping of Base Sequence in Frequency-Domain (occupying 11 consecutive tones)

Using a sampling frequency of 1.92MHz, an NPSS symbol should consist of 137 samples that can be obtained by size-128, zero-padded IFFT in addition to a CP of size-9, that applies to all symbols except the 5th symbol that will need to have a CP of size-10 to achieve full compatibility with legacy LTE and ordinary subframes [7]. Figure 9 illustrates the waveform.

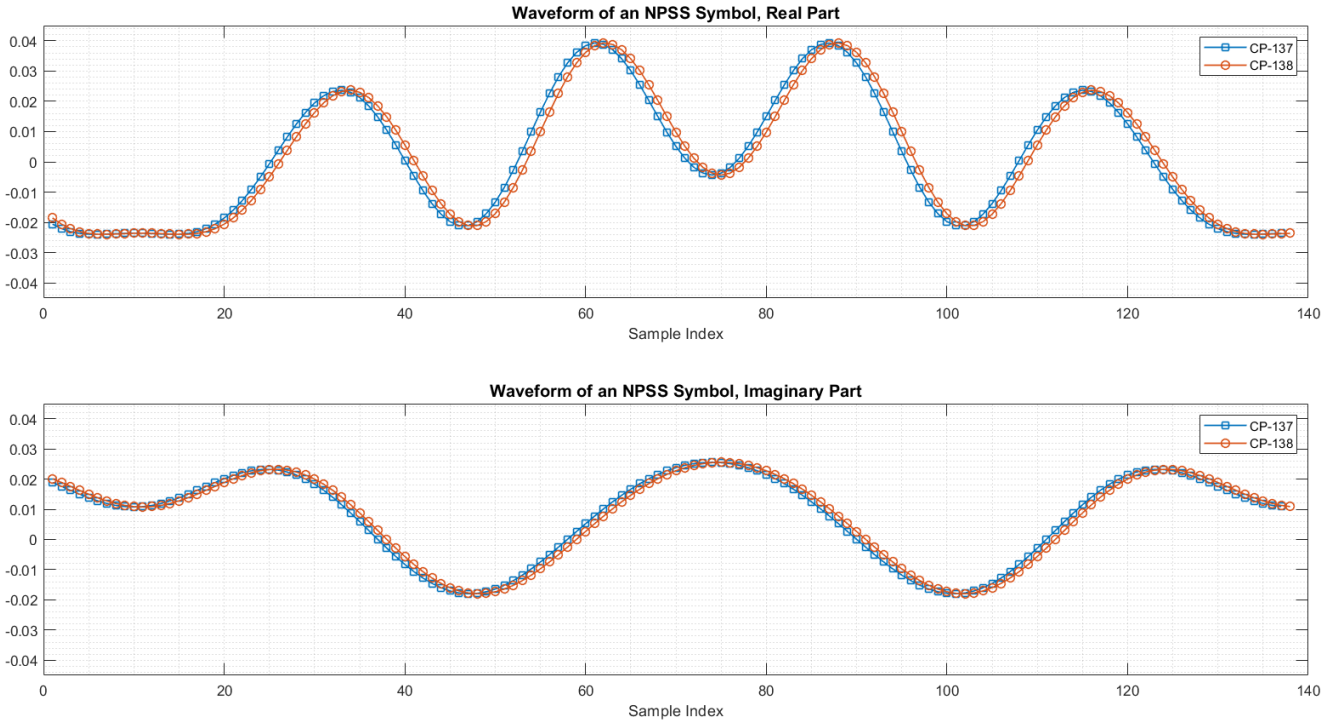


Figure 9 Waveform of NPSS Symbols Generated at 1.92 MHz by Size-128 IFFT

Functionality outline

From the previously mentioned givens, an outline of the functionality of the synchronization block can be constructed. In short, the block should make use of the NPSS periodicity and good correlation properties to estimate the time and frequency offsets with a reasonable accuracy that is discussed later, perform CP removal and down-sampling after the synchronization process is finished and provide information about symbol position to facilitate other chain operations or serve other higher-layer purposes.

Algorithm

The algorithm presented here is a three-step procedure. The first step is to use a reduced averaged time-domain auto-correlation metric to acquire a coarse timing and an FFO estimate, leveraging in the process the repetitive nature of the NPSS. The metric originally is supposed to have a length of 19200 samples, one frame length, as this is the period of the NPSS, however, the metric is reduced by a factor of 16 to reduce memory demands, and then it is averaged over frames to reduce the effect of AWGN. Figure 10 roughly summarizes the algorithm.

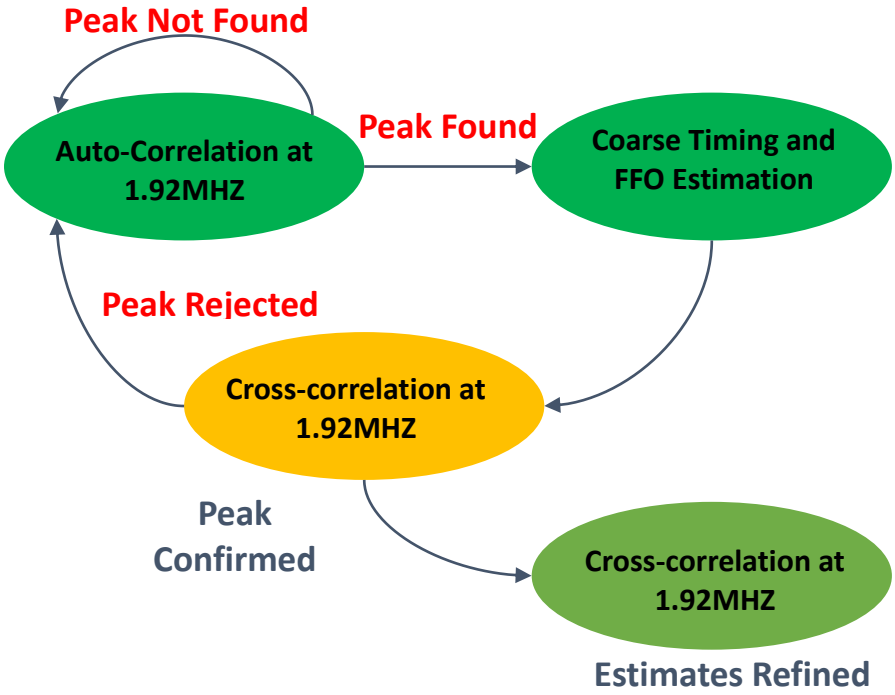


Figure 10 Algorithm High-Level State Diagram

Let $x(n)$ be a transmitted baseband OFDM signal, the received signal is given by,

$$r(n) = [x(n) * h(n)] \cdot e^{-j\frac{2\pi\varepsilon n}{N}} + w(n), \qquad N = 128, \qquad \varepsilon = \varepsilon_i + \varepsilon_f \tag{3}$$

Where $h(n)$ is the impulse response of the multipath channel, $w(n)$ represents AWGN, and $*$ denotes convolution. The frequency offset normalized to the subcarrier spacing is represented by ε and it has two components, integer frequency offset represented by ε_i, and fractional frequency offset represented by ε_f.

Now, the metric $R(k)$ calculated in a single frame period can be defined as,

$$R(k) = \sum_{i=k}^{k+N_w-1} \left(r(i) \cdot S\left(mod\left({}^{i}/_{N_s}, 11 \right) \right) \right) \cdot \left(r(i+N_s) \cdot S\left(mod\left({}^{i}/_{N_s} + 1, 11 \right) \right) \right)^* \tag{4}$$

Where k covers 19200 positions, N_w represents the auto-correlation window and it has a value of 1370, N_s represents the symbol length and hence has a value of 137 and $*$ here denotes complex conjugation. Then, $R_r(m)$ representing the reduced metric can be defined as,

$$R_r(m) = \sum_{i=16m}^{16(m+1)-1} R(i) \tag{5}$$

Where m covers 1200 positions only. Then the averaged metric $A(m)$ can be given by,

$$A(m) = A(m) \cdot (1-\alpha) + R_r(m) \cdot \alpha, \qquad 0 < \alpha < 1 \tag{6}$$

Now, coarse timing and FFO can jointly be estimated such that,

$$\tau = argmax\forall_m |A| \cdot 16 - 8 \tag{7}$$

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

However, this decision is only taken when,

$$\max(|A|) > Threshold \tag{9}$$

Where the value of this threshold is a design parameter. The second step in this procedure is to refine timing and estimate the IFO. It employs an NPSS matched filter to acquire these estimates.

Let us define the cross-correlation function, $C(\tau_c, \varepsilon_c)$ calculated in a single frame period,

$$C(\tau_c, \varepsilon_c) = \sum_{i=\tau_c}^{\tau_c+N_p-1} r(i) \cdot \left(p(i-\tau_c) \cdot e^{-j\frac{2\pi\varepsilon_c(i-\tau_c)}{N}}\right)^* \tag{10}$$

Where p is a locally generated NPSS, N_p is the length of the NPSS in samples, $\tau_c \in [\tau - \Delta, \tau + \Delta]$, and $\varepsilon_c \in [-2\ , 2] + \varepsilon_f$. The range of ε_i is chosen to cover the range mentioned earlier while Δ is considered a design parameter. This metric is averaged over a number of frames, considered a design parameter, and then if $\max(|C|) > Threshold$ the estimation is acquired such that,

$$(\tau, \varepsilon) = argmax\forall_{\tau_c, \varepsilon_c} |C| \tag{11}$$

However, if the maximum of the metric does not cross the threshold, another design parameter, the procedure goes back to the first step, and if the maximum passes, the procedure is completed normally. The third step is identical to the second, the difference only is that frequency hypotheses are taken around the current estimate with a chosen step to refine the estimates and reach a better accuracy. This step can be repeated. After finishing this procedure, the block switches to another mode of operation, where it removes the CP from incoming symbols, down-samples these symbols and passes their position information.

Constructing the Algorithm

It is of great importance to add this part to explain how algorithms in academic literature on the subject contributed to the algorithm presented here. Mainly, the algorithm presented earlier has the same structure of that presented in [7], but implements a different first stage very similar to that in presented in [11] except for the fact that the metric implemented here is a reduced one to save memory resources. There are two reasons behind this choice: the first is that the first stage in [11] is faster, and this was important as many like [11] and [12] emphasized speed as a means to reduce overhead power wasted by the RF stage during the synchronization process, the second reason is the relative simplicity of implementation in the first stage in [11]. This choice had its deteriorating effect on FFO estimation accuracy of the first stage, this posed a challenge. The third stage was added to equalize that effect and was guided mainly by [10] and to a less degree by [7]. Many techniques were inspired by the work in [7] like the peak finding process and the filtering process. The work in [12] deserves more attention in the future using a shared ram technique and possibly more resources, it was not implemented however as it needed more work and optimization to increase its frequency resolution and to cover the required range.

2.2. CFO correction [13:15]:

The aim of this block is to cancel the effect of the estimated carrier frequency offset that divided into two parts integer carrier frequency offset that come from synchronization block and fractional carrier frequency offset that come from fine synchronization block, from the symbol and this offset happen due to:

- Frequency difference between the transmitter and receiver oscillator.
- Oscillator instabilities.

Carrier frequency offset introduces ICI and destroy the orthogonality of sub-carrier so, it is important to correct it.

As the received signal $b(n)$ equation equals:

$$b(n) = \frac{1}{N}\sum_{k=0}^{N-1} A(k)G(k)e^{\frac{i2\pi(k+\varepsilon)n}{N}} + w(n) \quad ,n = 0,1,2,\dots.N-1 \tag{12}$$

Where A(k) is the modulated data, N is the number of sub-carriers used, G(k) is the channel gain at k^{th} sub-carrier, w(n) is the AWGN noise and ε is the frequency offset and it is equal:

$$\varepsilon = \varepsilon_{ICFO} + \varepsilon_{FCFO} \tag{13}$$

Where ε_{ICFO} is the integer carrier frequency offset and ε_{FCFO} is the fractional carrier frequency offset.

From equation (12) it's required to cancel the effect of the offset on the received symbol by multiplying the equation by exponential raised to phase conjugate to the estimated offset as following equation:

$$\hat{b}(n) = b(n)e^{\frac{-i2\pi(\varepsilon)n}{N}} \tag{14}$$

Where $\hat{b}(n)$ is the corrected symbol.

To achieve the effect of the conjugate exponential CORDIC algorithm is used to rotate the symbol by the required offset.

CORDIC algorithm (coordinate rotation digital computer)

CORDIC algorithm is an iterative algorithm for calculating trigonometric functions like sine, cosine and it is a hardware- efficient method which uses rotations to calculate a wide range of elementary functions using only shift and add. It has two modes of operations rotation mode (RM) and vectoring mode (VM).

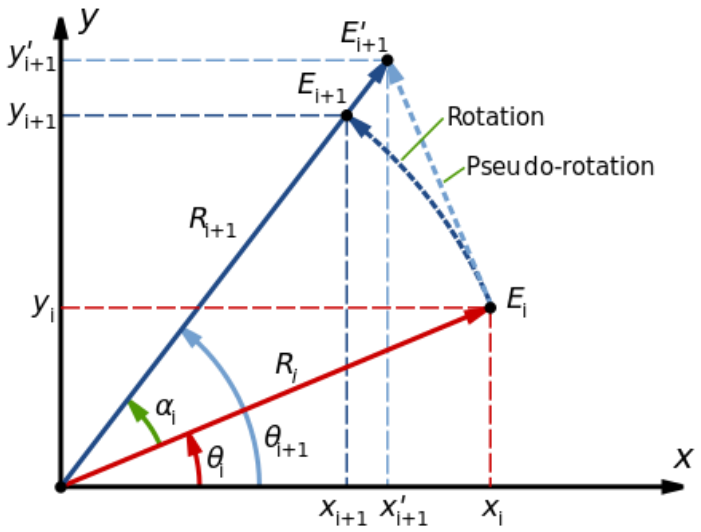


Figure 11 diagram show rotation of vector

Vector rotation can be generally calculated by following equations:

$$x_{i+1} = x_i \cos\alpha_i - y_i \sin\alpha_i \tag{15}$$

$$y_{i+1} = y_i \cos\alpha_i + x_i \sin\alpha_i \tag{16}$$

$$\theta_{i+!} = \theta_i + \alpha_i \tag{17}$$

By taking $\cos\alpha_i$ common factor we get:

$$x_{i+1} = \cos\alpha_i \,(x_i - y_i \tan\alpha_i) \tag{18}$$

$$y_{i+1} = \cos\alpha_i \,(y_i + x_i \tan\alpha_i) \tag{19}$$

$$\theta_{i+!} = \theta_i + \alpha_i \tag{20}$$

In order to simplify the implementation of algorithm $\tan\alpha_i$ is chosen to be equal 2^{-i} to make multiplication implemented as shift operations. So, the final form of equation after n iterations is:

$$x_{i+1} = (x_i - d_i y_i 2^{-i}) \tag{21}$$

$$y_{i+1} = (y_i + d_i x_i 2^{-i}) \tag{22}$$

$$\theta_{i+!} = \theta_i - d_i \alpha_i \tag{23}$$

Where d_i is the direction of rotation.

CORDIC in rotation mode:

$$x_{i+1} = (x_i - d_i y_i 2^{-i}) \tag{24}$$

$$y_{i+1} = (y_i + d_i x_i 2^{-i}) \tag{25}$$

$$\theta_{i+!} = \theta_i - d_i \alpha_i \tag{26}$$

And after n- iteration the equation will be:

$$x_n = K(x_0 \cos \alpha_0 - y_0 \sin \alpha_0) \quad (27)$$

$$y_n = K(y_0 \cos \alpha_0 + x_0 \sin \alpha_0) \quad (28)$$

$$\theta_n = 0 \quad (29)$$

The direction with each rotation takes obviously affects the accumulative angle that is rotated. Arbitrary angles can be rotated in the range $-99.7 \leq \theta \leq 99.7$. The sum of all angles obeying the law $\tan \alpha_i$ equal 2^{-i} is 99.7.

For angles outside this range trigonometric identities can be used to convert the desired angle into one within the range. The number of bits resolution in the angle is of course relevant to the final accuracy.

i	tanθ	Angle, θ	cosθ
1	1	45.0000000000	0.707106781
2	0.5	26.5650511771	0.894427191
3	0.25	14.0362434679	0.9701425
4	0.125	7.1250163489	0.992277877
5	0.0625	3.5763343750	0.998052578
6	0.03125	1.7899106082	0.999512076
7	0.015625	0.8951737102	0.999877952
8	0.0078125	0.4476141709	0.999969484
9	0.00390625	0.2238105004	0.999992371
10	0.001953125	0.1119056771	0.999998093
11	0.000976563	0.0559528919	0.999999523
12	0.000488281	0.0279764526	0.999999881
13	0.000244141	0.0139882271	0.99999997

cos 45 x cos 26.5 x cos 14.03 x cos 7.125 ... x cos 0.0139 = 0.607252941

From the previous table we get that as the number of iterations increase the accuracy of the algorithm increase and the constant multiplying by x_n and y_n nearly equal 0.60725941.

CORDIC in vectoring mode:

$$x_{i+1} = (x_i - d_i y_i 2^{-i}) \quad (30)$$

$$y_{i+1} = (y_i + d_i x_i 2^{-i}) \quad (31)$$

$$\theta_{i+!} = \theta_i - d_i \alpha_i \quad (32)$$

After n iterations the equations will equal:

$$x_n = K\sqrt{x^2 + y^2} \quad (33)$$

$$y_n = 0 \quad (34)$$

$$\theta_n = \theta_0 + \tan^{-1}\frac{y_0}{\theta_0} \quad (35)$$

2.3. FFT [16:31]:

The Fourier transform (FT) is used to transform a signal from the time domain to the frequency domain in order to analyze the signal's frequency components. In the frequency domain, the magnitude value for each frequency represents its contribution in the original signal, and the complex value shows the phase offset of the sinusoid at this frequency. The time domain signal is decomposed into a summation of multiple sinusoid signals, FFT is very useful conversion tool for processing a signal which is difficult to handle in the time domain, thus it is considered as one of the most important algorithms in digital signal processing and many communication systems. In order to compute the N point DFT it is required O (N^2) operations, however by using FFT, the required operations are down to O ($N\,log_2(N^2)$) [16][17][18].

In the mid-1960s, The Cooley-Tukey algorithm was introduced and it was a breakthrough in implementing Fast Fourier Transform (FFT) which reduce the complexity of a Discrete Fourier Transform (DFT). The DFT of a signal time domain signal x(n) as shown in equation [36].

$$x(k) = \sum_{k=0}^{n-1} x(n) W_N^{kn} \tag{36}$$

The $W_N^{kn} = e^{-j\frac{2\pi nk}{N}}$ is known as the twiddle factor and it is a complex value. Cooley-Tukey algorithm is a divide and conquer algorithm as it breaks the given problem into sub problems of the same size and solve these sub problems recursively. It combines the answers in at the end of the algorithm and generate the output. Number of stages to generate the output is log2N where N is the data size.

Algorithm:

FFT can be implemented using:

1- memory-based algorithm
2- pipeline algorithm

where pipeline classified to Multi-Path Delay Commutator (MDC) and single pass delay feedback (SDF).

2.3.1.1. Comparison between Memory based, SDF, and MDC

In memory based, the architecture consists of a unique butterfly unit that performs all the operations of the FFT, four two port RAMs to store the symbols, a memory with twiddle factors, address generators and control logic. In this algorithm, the data inputs from one memory are passed through the processing element to another memory and vice versa till the transform is completed, thus the latency in this algorithm is high.

In MDC architectures, the input sequence is divided into multiple parallel data streams by commutator then the data path through the butterfly and then multiplied by the twiddle factor.

For SDF architectures, a single data steam pass through the same multiplier and butterfly in each stage and the share the same storage elements. SDF architectures require minimum memory compared to MDC due to the efficient use of delay buffers. MDC architectures are not preferred for low power devices due to the large number of multipliers [17].

Table 1 Comparison of Pipelined FFT Architectures [17]

	Memory-based architecture	SDF architecture
Memory bank	r	$log_2 N$
Memory access times	2N$log_r N$	2N$log_2 N$
Complex multipliers	r-1	$log_r N$-1
Complex adders	2r	$2log_2 N$
Clock frequency	$log_r N$/r	1
Clock cycle	N$log_r N$/r	N

To achieve a memory-based architecture needs to drive its clock log_r(N/r) times the processor frequency to achieve the same performance as pipeline SDF architecture. So, pipeline architectures are the best choice when power and performance are the main concern than complexity. While memory-based architectures are good choice when the complexity is the main concern [17]. Thus, SDF is implemented without parallelization to minimize the power consumption.

The radix, r, stands for the number of parts that the input signal will be divided into. There are a lot of radix can be used with 16-point FFT such as Radix 2, Radix 4, Split Radix, Radix 2^2

2.3.1.2. Comparison between Radix 2, Radix 4, Split Radix, Radix 2^2

i. Radix2

RAD2 DIF algorithm is obtained by using the divide and conquer approach to the DFT problem. To compute the DFT, first split Equation [36] into two summations, one of which involves the sum over the first data point while the other over the next data points as shown in Equation [37]

$$x(k) = \sum_{k=0}^{\frac{N}{2}-1} x(n) \cdot W_N^{kn} + \sum_{k=\frac{N}{2}}^{N-1} x(n) \cdot W_N^{kn} \tag{37}$$

$$\because W_N^{k\frac{N}{2}} = (-1)^k \tag{38}$$

$$\therefore\ x(k) = \sum_{k=0}^{\frac{N}{2}-1} (x(n) + (-1)^k x\left(n + \frac{N}{2}\right)) \cdot W_N^{kn} \tag{39}$$

Considering the even and odd-numbered frequency samples separately results in

$$x(2k) = \sum_{k=0}^{\frac{N}{2}-1} (x(n) + x\left(n + \frac{N}{2}\right)) \cdot W_{\frac{N}{2}}^{kn} \tag{40}$$

$$\therefore\ x(2k+1) = \sum_{k=0}^{\frac{N}{2}-1} ((x(n) - x\left(n + \frac{N}{2}\right)) \cdot W_{\frac{N}{2}}^{kn}) \cdot W_{\frac{N}{2}}^{kn} \tag{41}$$

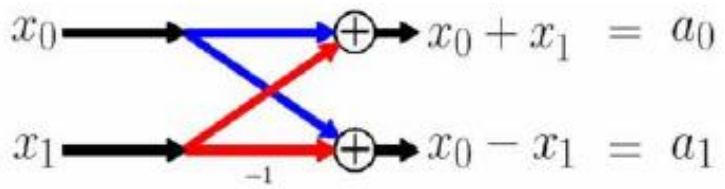


Figure 12 Radix-2 butterfly unit [25]

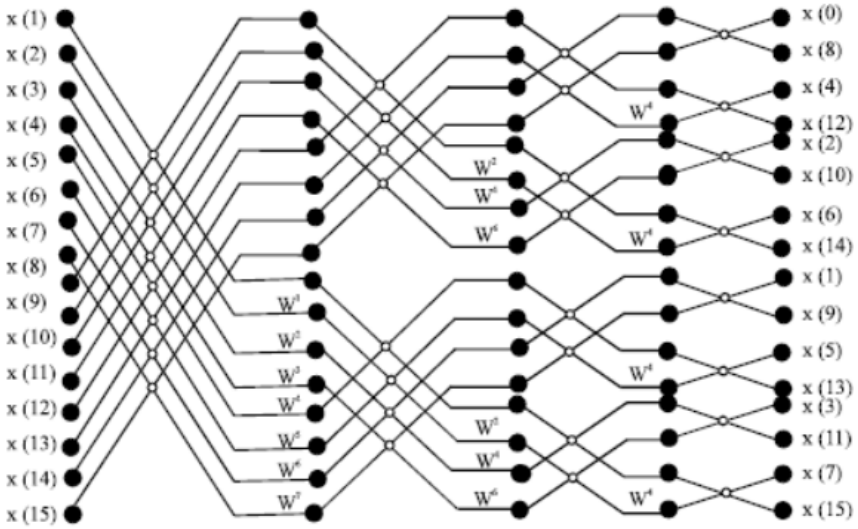


Figure 13 Flow graph of 16-point Radix-2 FFT algorithm

As the output is out-of-ordered, then a bit reversed operation is required to place the frequency samples in correct place. In RAD2 there is $Nlog_2(N)$ complex addition and $\frac{N}{2}log_2(N)$ complex multiplication.[16]

ii. Radix4

RAD4 algorithm is similar to RAD2, however instead of splitting the DFT computation into halves in RAD2, it splits the DFT into four. Thus, the N-point input sequence is split into four subsequences, x(4n), x(4n+1), x(4n+2), x(4n+3) where n=0,1, ….. , $\frac{N}{4}-1$

$$x(k)=\sum_{k=0}^{\frac{N}{4}-1} x(n)\cdot W_N^{kn}+\sum_{k=\frac{N}{4}}^{\frac{N}{2}-1} x(n)\cdot W_N^{kn}+\sum_{k=\frac{N}{2}}^{\frac{3N}{4}-1} x(n)\cdot W_N^{kn}+\sum_{k=\frac{3N}{4}}^{N-1} x(n)\cdot W_N^{kn} \tag{42}$$

$$\therefore x(l,q)=\sum_{m=0}^{\frac{N}{4}-1} x(l,m)\cdot W_{N/4}^{kn} \tag{43}$$

$$x(p\,,q)=x(\frac{N}{4}\cdot p+q) \tag{44}$$

$$x(l,m)=x(4m+l) \tag{45}$$

$$where\ l,q=0,1,2,3\ and\ m,q=0,1,\ldots.\frac{N}{4}-1$$

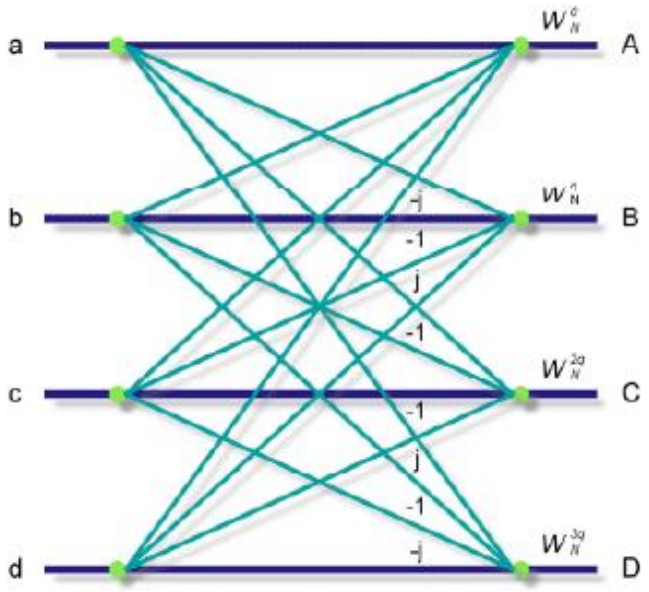


Figure 14 Radix-4 butterfly unit [26]

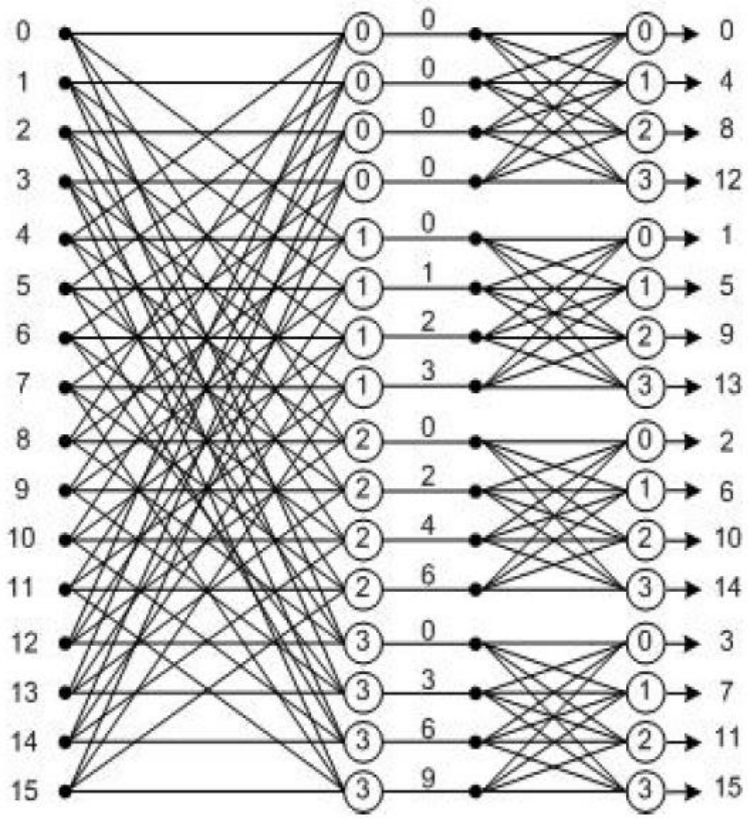


Figure 15 Flow graph of 16-point Radix-4 FFT algorithm [26]

As the output is out-of-ordered, then a bit reversed operation is required to place the frequency samples in correct place. In RAD4 butterfly there is 8 complex addition and 3 complex multiplication and for 16-point FFT there is 96 complex addition and 8 complex multiplication [24][25] [26][27].

iii. Split Radix

The SRFFT algorithm is based on the synthesis of one half-length DFT together with two quarter-length DFTs. This is possible because, in the RAD2 computations, the even-indexed points can be computed independent of the odd-indexed points. The SRFFT algorithm uses the RAD4 algorithm to compute the odd-numbered points. Hence, the N-point DFT is decomposed into one N/2-point DFT and two N/4-point DFTs

$$x(2k) = \sum_{n=0}^{\frac{N}{2}-1} (x(n) + x(n + N/2))W^{4kn} \tag{46}$$

$$x(4k+3) = \sum_{n=0}^{\frac{N}{4}-1} (g(n) + jf(n))W^{3n} \cdot W^{4kn} \tag{47}$$

$$x(4k+1) = \sum_{n=0}^{\frac{N}{4}-1} (g(n) - jf(n))W^n \cdot W^{4kn} \tag{48}$$

$$g(n) = x(n) - x\left(n + \frac{N}{2}\right) \tag{49}$$

$$f(n) = x(n + N/4) - x\left(n + \frac{3N}{4}\right) \tag{50}$$

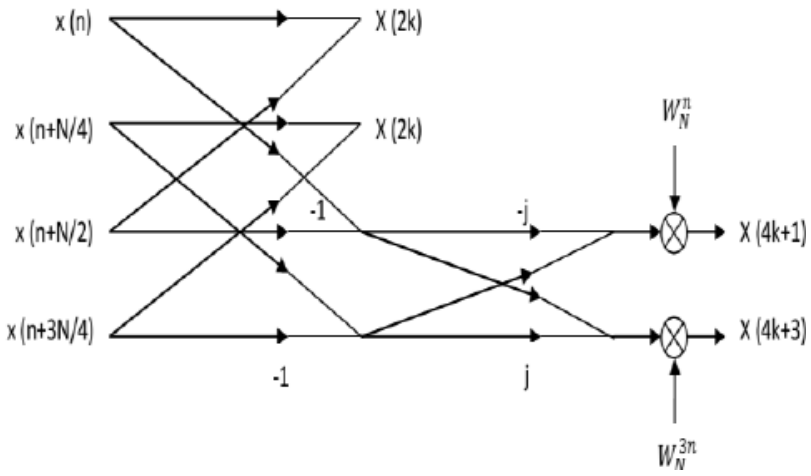


Figure 16 Split-radix butterfly unit [29]

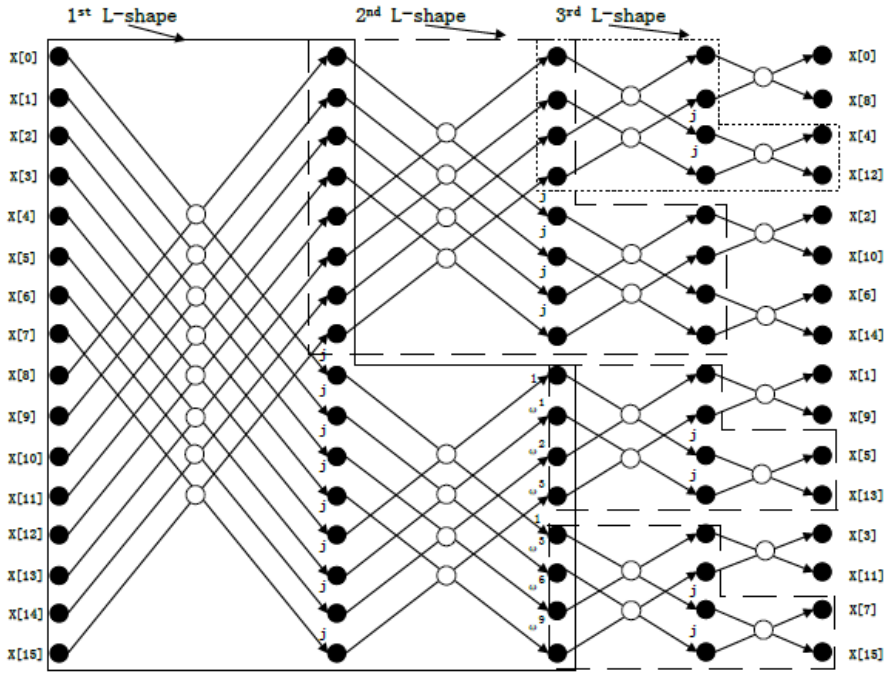


Figure 17 Flow graph of 16-point SPLIT Radix FFT algorithm [30]

As the output is out-of-ordered, then a bit reversed operation is required to place the frequency samples in correct place. For 16-point FFT there is 64 complex addition and 8 complex multiplication.
SRFFT has the same number of adders as in Radix 2 and same number of multipliers as in Radix4 so it is good architecture in terms of low power consumption. However, its irregular design makes it not the best choice for RTL implementation [28][29][30].

iv. Radix 2^2

Radix 2^2 was developed to inherit the radix 2 in terms of its simple control structure and it has the advantage of saving hardware in other words reducing the number of multipliers used as in radix 4. The Discrete Fourier Transform (DFT) of N-point input x(n) is defined as

$$x(k) = \sum_{n=0}^{N-1} x(n) \cdot W_N^{nk} \qquad 0 \le k < N \quad \text{where } W_N = e^{-j\frac{2\pi}{N}} \tag{51}$$

$$x(k_1 + 2k_2 + 4k_3) = \sum_{n_{3=0}}^{\frac{N}{4}-1} H(k_1, k_2, k_3) \cdot W_N^{n_{3(k_1+2k_2)}} W_{N/4}^{n_{3(k_3)}} \tag{52}$$

$$\mathrm{H}(k_1, k_2, n_3) = [\mathrm{x}(n_3) + (-1)^{k_1} \cdot \mathrm{x}(n_3 + \tfrac{N}{2})] + (-\mathrm{j})^{(k_1+2k_2)} [\mathrm{x}(n_3 + \tfrac{N}{4}) + (-1)^{k_1} \cdot \mathrm{x}(n_3 + \tfrac{3N}{4})] \tag{53}$$

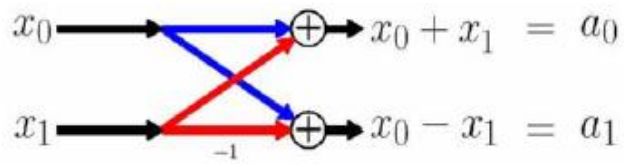


Figure 18 Radix-2^2 butterfly unit [25]

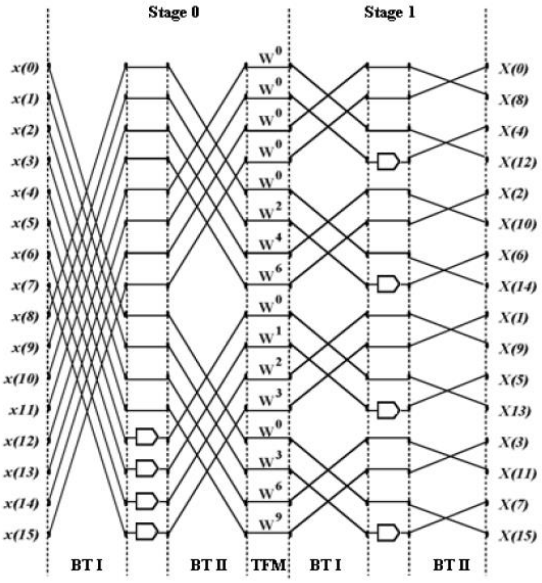


Figure 19 Flow graph of 16-point Radix-2^2 FFT algorithm [20]

$$\begin{bmatrix} S1[0] \\ S1[1] \\ S1[2] \\ S1[3] \\ S1[4] \\ S1[5] \\ S1[6] \\ S1[7] \end{bmatrix} = \begin{bmatrix} I[0]+I[8] \\ I[1]+I[9] \\ I[2]+I[10] \\ I[3]+I[11] \\ I[4]+I[12] \\ I[5]+I[13] \\ I[6]+I[14] \\ I[7]+I[15] \end{bmatrix} \quad \begin{bmatrix} S1[8] \\ S1[9] \\ S1[10] \\ S1[11] \\ S1[12] \\ S1[13] \\ S1[14] \\ S1[15] \end{bmatrix} = \begin{bmatrix} I[0]-I[8] \\ I[1]-I[9] \\ I[2]-I[10] \\ I[3]-I[11] \\ (I[4]-I[12]) \times -J \\ (I[5]-I[13]) \times -J \\ (I[6]-I[14]) \times -J \\ (I[7]-I[15]) \times -J \end{bmatrix}$$

Stage 1 16-point FFT arithmetical operations

$$\begin{bmatrix} S2[0] \\ S2[1] \\ S2[2] \\ S2[3] \\ S2[4] \\ S2[5] \\ S2[6] \\ S2[7] \end{bmatrix} = \begin{bmatrix} (S1[0]+S1[4]) \\ (S1[1]+S1[5]) \times (0.9239 - J0.3827) \\ (S1[2]+S1[6]) \times (0.7071 - J0.7071) \\ (S1[3]+S1[7]) \times (0.3827 - J0.9239) \\ (S1[0]-S1[4]) \\ (S1[1]-S1[5]) \times (0.3827 - J0.9239) \\ (S1[2]-S1[6]) \times (-0.7071 - J0.7071) \\ (S1[3]-S1[7]) \times (-0.9239 + J0.3827) \end{bmatrix} \quad \begin{bmatrix} S2[8] \\ S2[9] \\ S2[10] \\ S2[11] \\ S2[12] \\ S2[13] \\ S2[14] \\ S2[15] \end{bmatrix} \begin{bmatrix} (S1[8]+S1[12]) \\ (S1[9]+S1[13]) \times (0.9239 - J0.3827) \\ (S1[10]+S1[14]) \times (0.7071 - J0.7071) \\ (S1[11]+S1[15]) \times (0.3827 - J0.9239) \\ (S1[8]-S1[12]) \\ (S1[9]-S1[13]) \times (0.3827 - J0.9239) \\ (S1[10]-S1[14]) \times (-0.7071 - J0.7071) \\ (S1[11]-S1[15]) \times (-0.9239 + J0.3827) \end{bmatrix}$$

Stage 2 16-point FFT arithmetical operations

$$\begin{bmatrix} S3[0+n] \\ S3[1+n] \\ S3[2+n] \\ S3[3+n] \end{bmatrix} = \begin{bmatrix} (S2[0+n]+S2[2+n]) \\ (S2[1+n]+S2[3+n]) \\ (S2[0+n]+S2[2+n]) \\ (S2[1+n]+S2[3+n]) \times -J \end{bmatrix}, \text{ n=0,4,8,12}$$

Stage 3 16-point FFT arithmetical operations

$$\begin{bmatrix} S4[0+n] \\ S4[1+n] \end{bmatrix} = \begin{bmatrix} (S3[0+n]+S3[1+n]) \\ (S3[0+n]-S3[1+n]) \end{bmatrix}, \text{ n=0,2,4,6,8,10,12,14}$$

Stage 4 16-point FFT arithmetical operations

As the output is out-of-ordered, then a bit reversed operation is required to place the frequency samples in correct place. For 16-point FFT there is 64 complex addition and 8 complex multiplication.

Radix 2^2 is the best choice for low power design and it is very popular with pipeline FFT implementation to reduce the power consumption [20][21][22][23][31]

Table 2 FFT different radixes comparison

RADIX	#complex addition	#complex multiplication	Complexity
RAD2	64	17	Simple
RAD4	96	9	Complex
SPLIT RADIX	64	8	Simple
RADIX 2^2	64	8	Simple

All the algorithms' have been implemented using MATLAB. RAD2 and RAD4 are not the best choice when it comes to area and power, also the split radix has irregular design which make it not suitable for digital design. Therefore, RADIX 2^2 is the one which is implemented in RTL as it is the best in terms of area and power. Moreover, we implement the algorithm using SDF and using some optimization techniques, to minimize the power, area, and latency.

2.4. RESOURCE DE-MAPPER [32]:

Resource block is used to represent the mapping of certain physical channels to resource elements. $N_{\text{symb}}^{\text{DL}}$ consecutive OFDM symbols in the time domain and $N_{\text{sc}}^{\text{RB}}$ consecutive subcarriers in the frequency domain are used to define the resource clock because the resource block consists of $N_{\text{symb}}^{\text{DL}} \times N_{\text{sc}}^{\text{RB}}$ resource elements, corresponding to one slot in the time domain and 180 kHz in the frequency domain. Physical resource blocks are numbered from 0 to $N_{\text{RB}}^{\text{DL}} - 1$ in the frequency domain. [5][6]

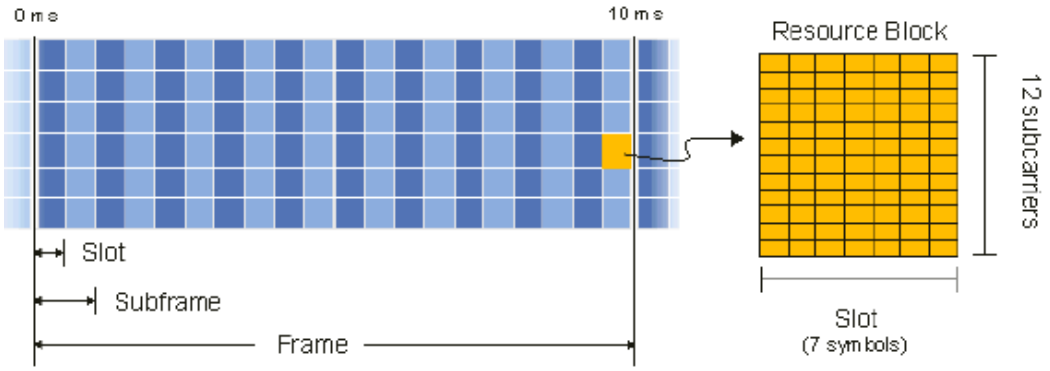


Figure 20 Resource block [4]

Resource block pair consists of two resource blocks the even numbered frame and the odd numbered frame. As in NB-IOT the Normal cyclic prefix is used then the $\Delta f = 15\ kHz$, $N_{\mathrm{sc}}^{\mathrm{RB}}$=12, and $N_{\mathrm{symb}}^{\mathrm{DL}}$=7. Then the subframe is 12×14 and only one antenna port used. Then the NB-IOT carrier uses one LTE PRB in the frequency domain where there is twelve 15kHz subcarriers for a total of 180kHz.[5][6]

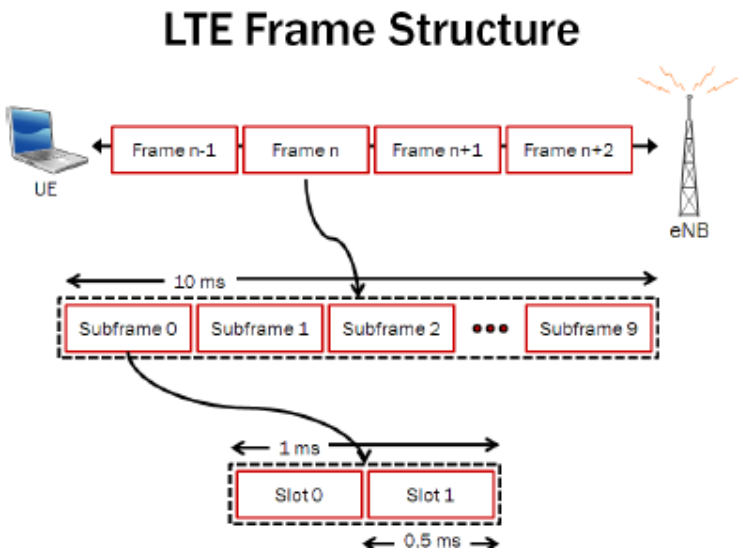


Figure 21 LTE Frame Structure [4]

Each frame is 10ms and it consists of 10 subframe each subframe is 1ms in time access. Each subframe consists of two slots where each slot is 0.5 msec. Slot is known as resource block and the resource block contains 7 OFDM symbols [32]

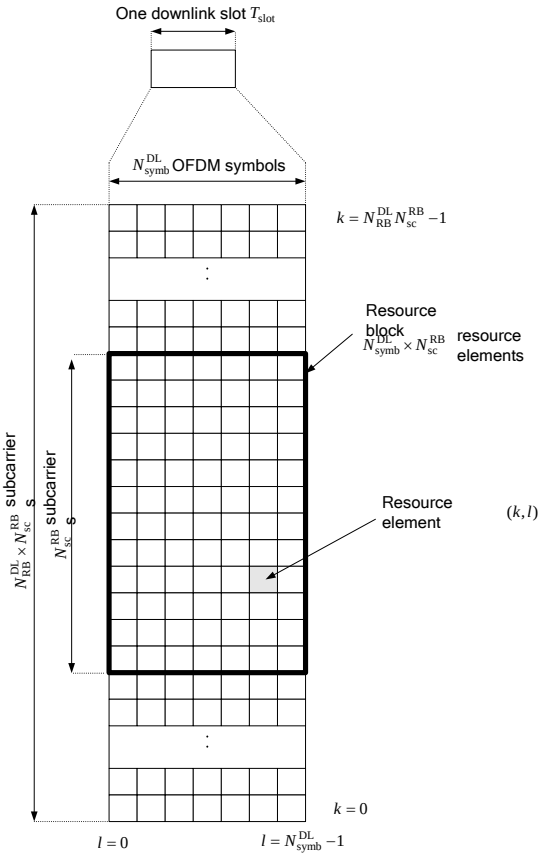


Figure 22 Downlink resource grid

2.5. Channel Estimation [33:38]:

Channel Model for LTE and NB-IoT.

- Channel Models:
  - Static propagation
    Static Channel model present the channel as an additive white Gaussian noise to the signal with power related to the SNR and no multi-path propagation or fading took into consideration for this model
  - Multi-path fading propagation

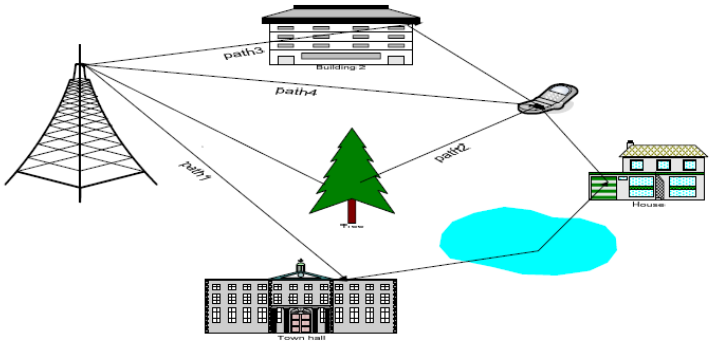


Figure 23 Multi-path fading channel model

Multi-path fading channel model present the channel with multi-path fading and Doppler shift there are many different environments differ from each other in the delay profile.

The delay profile means that The receiver found multipath signals with different propagation delays called **Taps**, these copies either make constructive or destructive interference as found in figure 22 there are three paths for the signals with different delays due to different obstacles as shown in the power delay profile thus two classifications can be describe the wireless channel environment:

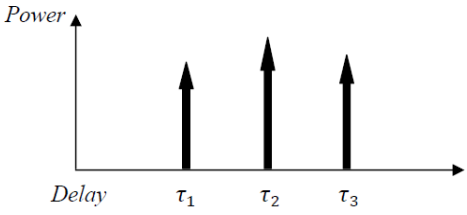


Figure 24 Delay profile

- Large-scale fading:
  Determine the large variations found in the received signal amplitude or the power level may cause through:
  - Path loss: The received signal strength decreases over large distance between the transmitter and the receiver.
  - Shadowing: Caused by some obstacles found along the path between transmitter and receiver.

- Small-scale fading:
  A small variation happened in phase and amplitude caused by interference between versions of these paths due to moving of either transmitter or the receiver (Doppler effect) or nearby reflecting wall.

- High-speed train condition
  High-speed train channel model presents the channel to simulate the high-speed movement of the UE (user Equipment) but without fading and an AWGN could be added.

- Specification of Used Channel Model
  The multi-path fading channel is used in our project to simulate the channel model and get the performance of the chain when the medium between the transmitter and the receiver follow the multi-path fading channel model

  Specifically, the Channel model used in our project is, Extended typical Urban model (ETU)

  As shown from (figure3) taken from the 3GPP standard, that is the delay profile of our multi-path fading channel model

  Specification for the channel model used in simulation is:

  Rayleigh fading channel with an Extended Typical Urban delay profile, maximum Doppler frequency shift equal 5 Hz, and with sampling frequency equal 1.92 MHz.

Excess tap delay [ns]	Relative power [dB]
0	-1.0
50	-1.0
120	-1.0
200	0.0
230	0.0
500	0.0
1600	-3.0
2300	-5.0
5000	-7.0

Figure 25 Delay profile for ETU according to 3GPP

Types of channel estimation

Channel estimation process used in compensate the effect of the channel on the transferring signals and removing inter symbol interference and noise that affect the signal

Channel Estimation can be grouped into three categories as pilot based, blind and semi-blind.

- Pilot based Channel Estimation, some of data symbols are used to estimate channel.
- Blind Channel Estimation statistical properties of channel are used.
- Semi-blind Channel Estimation, take information about channel from data symbols and statistical properties, so both used in the channel estimation process.

Narrowband-IoT and LTE use Orthogonal Frequency Division Multiplexing (OFDM) scheme

The channel Estimation in Narrow band Physical Downlink Shared Channel (NPDSCH) use Pilot-based channel estimation throughout putting some known symbols at the transmitter and generate the same symbols at the receiver to find the relation between the received and the original symbols and get a rough estimation for the channel.

These pilots called Narrow band Resource Elements (NRS). Their values and locations calculated using some equations and sequence generators. They are found also inside all the sub frames transmitted except the Narrow band primary synchronization signals (NPSS) and Narrow band secondary synchronization signals (NSSS).

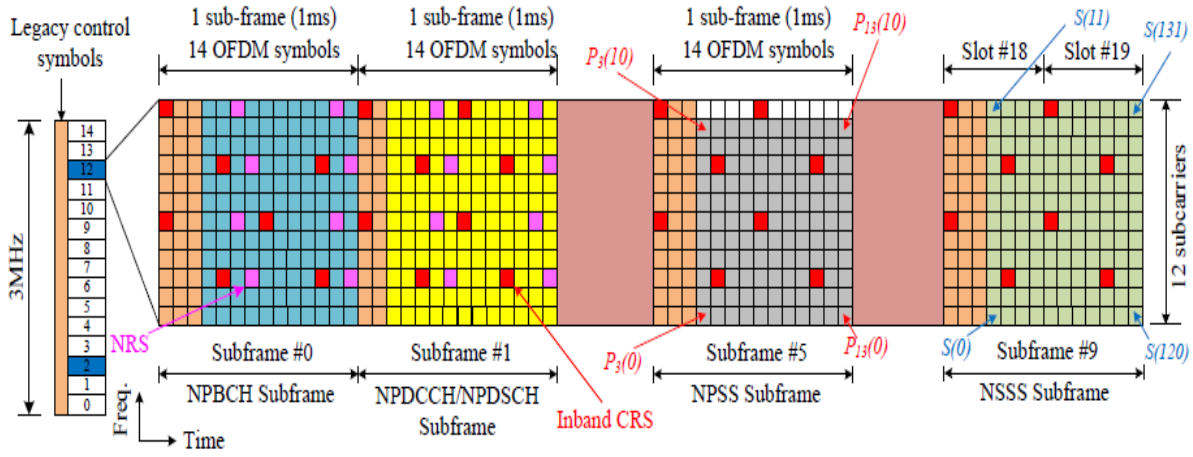


Figure 26 NB-IoT frame transmitted in an in-band mode over an LTE frame

This figure illustrates the distribution of the pilots over a NB-IoT frame transmitted in in-band mode consists of 20 slots, each two slots formed one sub-frame with 12 subcarrier frequencies and 14 symbols in frequency and time respectively.

It is obvious that pilots denoted by NRS are found in all the subframes except the NPSS, and NSSS sub-frames, while CRS is referred to the Cell specific Reference Signal that related to the LTE structure.

Channel Estimation Algorithms

There are many channel estimation algorithms used in NB-IoT receiver but the problem appears in form of a tradeoff between high complexity with high performance of the design and the required low power and area.

From these algorithms, there is Linear Minimum Mean Square error (LMMS) Algorithm with high performance but the complexity of the design also high so Least Square channel estimation algorithm is preferred to use in our chain and to compensate the performance we use interpolation process to get better performance with low complexity.

Least Square (LS) Channel Estimation

After the channel effect assuming the input to the receiver is y:

$$y = xH + n$$

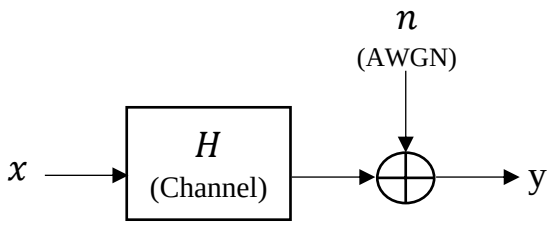


Figure 27 Channel effect on transmitted signal

x: The required transmitted signal.
H: The channel effects.
n: Additive White Gaussian Noise.
y: The received signal.
Least Square channel estimation algorithm:

$$H_{LS} = x^{-1}y$$

LS algorithm ignores the effect of the AWGN so its performance is low but also with low complexity.

Channel Estimation Block aims to divide the received pilots over the transmitted pilots to get the channel effect and then the equalizer divided the channel effect over all the transmitted symbols to compensate the effect of the channel for all the transmitted symbols.

Interpolation

From the previous explanation it is obvious that there should be an interpolation process after the estimation process as pilots found in four only sub carrier but the sub frame contains twelve subcarriers

Therefore, there are three different case for the interpolation process after Channel Estimator:

1. The zero-order interpolation:

In this case the interpolation is constant so each three subcarriers contain one pilot divide by the same channel H_{LS}
In each slot the Channel Estimation, get four different outputs H_{LS} each one used by the equalizer with three consecutive sub-carriers.
This design is simpler with less hardware but less performance and the variation of Channel Estimation output happened each slot.

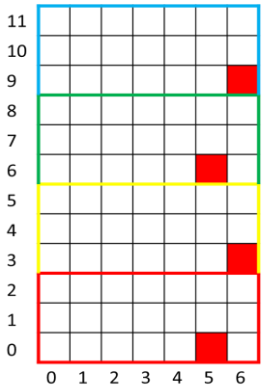


Figure 28 Zero-order interpolation

2. Interpolation over each slot:

In this case the interpolation is linear and calculated every slot

In each slot the Channel Estimation, get twelve different outputs (H_{LS}) each one used by the equalizer with one of the twelve sub-carriers.

This design in more complex than the previous with extra hardware, but Small variations in performance compared to the previous, and variation of Channel Estimation output happened each slot.

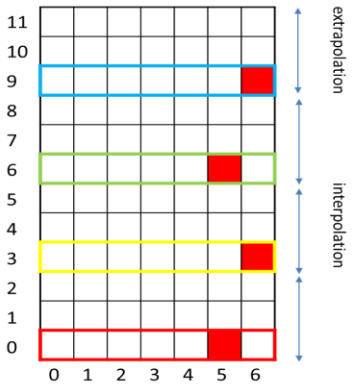


Figure 29 Slot interpolation

3. Interpolation over Sub-frame:

In this case, the interpolation is linear and calculated every Sub-Frame.

In each Sub-Frame the Channel Estimation, get twelve different outputs (H_{LS}) each one used by the equalizer with one of the twelve sub-carriers.

This design is more complex than the previous with more hardware but have great impact on the performance compared to the previous, and variation of Channel Estimation output happened each sub-frame.

Therefore, it considered the optimal design to achieve the best performance with the optimal hardware.

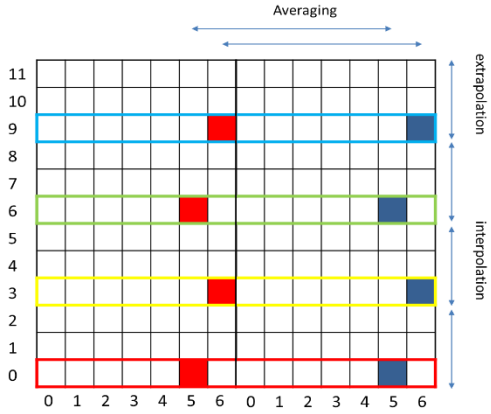


Figure 30 Sub-Frame interpolation

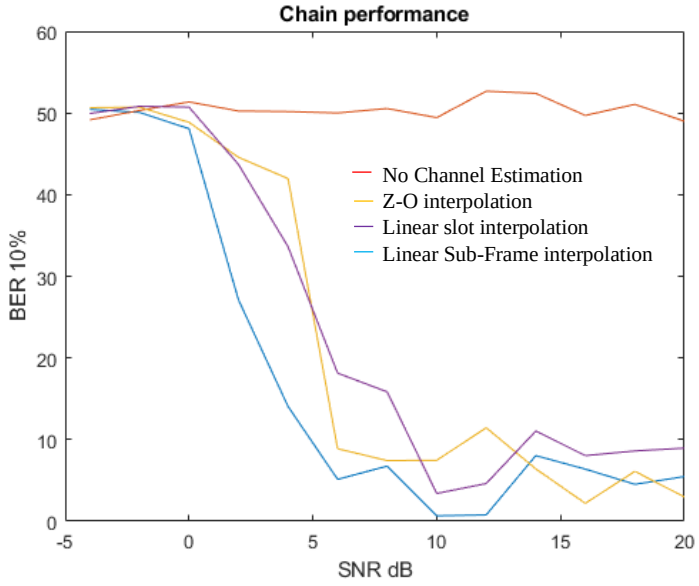


Figure 31 BER vs SNR Matlab simulation for the three designs to get a rough estimation about the best performance

2.6. NRS Value Generation:

NRS stands for Narrow band Reference Signals that acts as pilots, which help in channel estimation process to get an estimate for the channel and equalize the effect of this channel. These pilots found inside each Sub-frame except two Sub-frames NPSS and NSSS Sub-frame.

According to 3GPP, there are specific equation to get out the values of these pilots and other equations to get the Location of these NRS signals.

Theoretical Method to get NRS values:

NRS value depends on three variables $l, n_s, and\ m$ so that the value of NRS changes according to any variation of these parameters.

$l: denote\ for\ the\ number\ of\ symboles\ that\ contain\ the\ NRS\ equal\ 5,6$

$n_s: denote\ for\ the\ number\ of\ the\ recieved\ slot\ its\ range\ [0,9]$

$m = 0,1,\cdots,2N_{RB}^{\max DL} - 1 \quad for\ narrow\ band\ N_{RB}^{\max DL} = 1\ as\ there\ are\ only\ one\ resource\ block$

$so\ m = 0,1$

$$NRS\ values\ equation: r_{l,n_s}(m) = \frac{1}{\sqrt{2}}\big(1 - 2c(2m)\big) + j\frac{1}{\sqrt{2}}\big(1 - 2c(2m+1)\big) \tag{54}$$

$C: is\ refer\ to\ pseudo - random\ sequence\ generation\ defined\ by\ a\ length - 31\ Gold\ Sequence$

$$C\ equation: c(n) = \big(x_1(n + N_c) + x_2(n + N_c)\big) mod 2 \tag{55}$$

$$x_1(n + 31) = \big(x_1(n + 3) + x_1(n)\big) mod 2 \tag{56}$$

$$x_2(n + 31) = \big(x_2(n + 3) + x_2(n + 2) + x_2(n + 1) + x_2(n)\big) mod 2 \tag{57}$$

The initialization of the first m-sequence is constant so

$$x_1\ initialized\ with\ x_1(0) = 1, and\ x_1(n) = 0, n = 1,2,\cdots,30 \tag{58}$$

While the second m-sequence depends on the application so for NRS generation x_2 initialized with

$$C_{init} = 2^{10}(7(n_s + 1) + l + 1)\big(2N_{ID}^{cell} + 1\big) + 2N_{ID}^{cell} + N_{CP} \tag{59}$$

$where\ N_c = 1600,\ N_{ID}^{cell}\ is\ an\ input,\ and\ N_{CP} = 1\ for\ normal\ CP.$

2.7. NRS Location Generation:

As Narrowband Reference Signals have equations to get their values, also there are equations to get the places where these pilots found inside the resource block grid.

These places generated at the transmitter and at the receiver by the same values to be same.

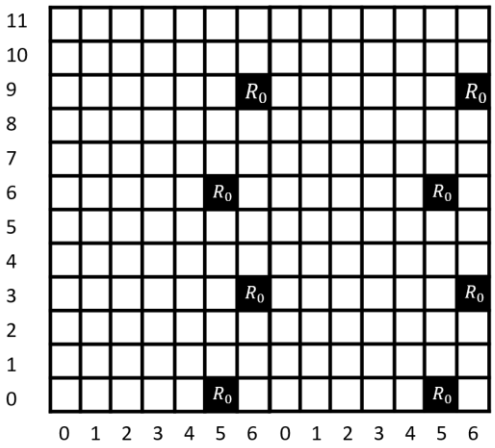


Figure 32 R0: refer to the resource element placed at $N_{ID}^{cell} = 0$

Theoretical Method to get NRS values:

NRS Location depends on three variables $v, v_{shift}, and\ m$

$NRS\ Location\ equation$:

$$k = 6m + \left(v + v_{shift}\right) mode 6 \tag{60}$$

$$l = N_{symb}^{DL} - 2,\ \ N_{symb}^{DL} - 1 = 5,6\ in\ each\ slot \tag{61}$$

k: refer to the place of a pilot to be in which row, ranged from 0 to 11 as row refer to the subcarrier

l: refer to the symbol number, ranged from 0 to 6.

N_{symb}^{DL}: refer to number of symbols in one slot for downlink and equal to 7.

$$v = \begin{cases} 0 & if\ l = 5 \\ 3 & if\ l = 6 \end{cases} \tag{62}$$

$$v_{shift} = N_{ID}^{cell} mod 6 \tag{63}$$

$$m = 0,1,\cdots,2N_{RB}^{\max DL} - 1 \quad for\ narrow\ band\ N_{RB}^{\max DL} = 1\ as\ there\ are\ only\ one\ resource\ block \tag{64}$$
$$so\ m = 0,1$$

2.8. Channel Equalizer:

In digital communication wireless system, data transmitted from transmitter through channel to the receiver and the data affected by some noise and multipath fading problems so the received data at receiver can expressed by the following equation:

$$r_k = s_k \otimes h_k + n_k \quad (65)$$

Where r_k is the received data, h_k represent the channel and $n_k$represent the effect of noise which is usually AWGN. This equation is in time domain.

In order to receive all symbols, correct as possible we need to know the effect of channel to cancel it. So, after the effect of channel is estimated for all frequencies in the resource block, equalizer needs to cancel this estimated effect from all symbols by dividing each symbol by the corresponding value of channel estimated.

After equalization the received data can be expressed in frequency domain by the following equation:

$$y_k = s_k * \frac{h_k}{h_{eq}} + \frac{n_k}{h_{eq}} \quad (66)$$

Where h_{eq} is the estimated value of channel.

In channel equalizer block it is required to divide the channel contribution that the channel estimator estimates by the symbol that get out from resource de-mapper block. So, the block depending on complex divider algorithm.

Complex divider algorithm.

We can implement complex divider block directly but it isn't commonly used as it take large area and power and complex design so, the commonly used block used is the complex multiplier so, let's see how can we use it to perform the desired function.

For example, we want to divide $a + ib$ by $c + id$ $(\frac{a+ib}{c+id})$ if we multiply the numerator and denumerator by the complex conjugate of denumerator $(c - id)$ the result will be $\frac{(ac+bd)+j(bc-ad)}{c^2+d^2}$ the numerator of the result can be out from a complex multiplier between the symbol and complex conjugate of estimated channel and the denumerator is a real positive number which is only a scaling factor can be done by real divider or the scaling factor can be neglected without having any degradation in the accuracy of the chain as the de-modulator block only need the sign of the real and imaginary to detect the true bits as it works with hard decision.

2.9. Fine Synchronization [39:40]:

2.9.1. Problem Definition:

After coarse estimation phase the rule of this block ends, but there are some problems and mismatches are made on the chain such as the channel effect and oscillator effect. So, for this purpose we make use of NB reference signals (NRSs) for tracking time and frequency offsets. NRS signals are distributed across frequency and time grid as shown in Figure 33.

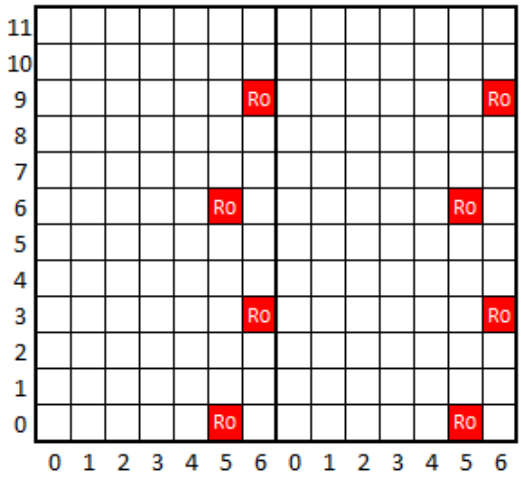


Figure 33 Example for NRS location in subframe

Our used algorithm (Kiran Kuchi, 2019) achieve significant detection performance even at low SNR values like -12 dB which is typical operation range of low power NB-IoT devices [40].

Algorithm:

2.9.1.1. Residual Time Offset (RTO):

As shown in Figure 33 NRSs signals in an OFDM symbol are seperated by 90 kHz. And the frequency. As the frequency response of a channel with high delay spreads varies across 6 subcarriers. So NRS signals of two consecutive OFDM symbols is considered to track timing offsets.

We apply conjugate product on one consecutive pair of NRSs and compare the product with the conjugate product of the consecutive pair of NRSs generated by NRSs generation block as shown in equation [67]

$$\frac{R_l(p)R_{l+1}^*(p+v)}{X_l(p)X_{l+1}^*(p+v)} = K_1 e^{\frac{-j2\pi(\tau_r v-\varepsilon_r(N+N_{CP}))}{N}} \quad (67)$$

Then we repeat this equation but for between the non-consecutive pilots as shown in equation [68]

$$\frac{R_l^*(p+N_s-1)R_{l+1}(p+v)}{X_l^*(p+N_s-1)X_{l+1}(p+v)} = K_2 e^{\frac{-j2\pi(\tau_r v+\varepsilon_r(N+N_{CP}))}{N}} \quad (68)$$

By solving the two equations and eliminating the effects of residual frequency offset (RFO), estimate of residual time offset (RTO) is obtained as in equation 69

$$\tau_r = \frac{N}{4\pi|v|}\left[\angle\left(\frac{R_l(p)R_{l+1}^*(p+v)}{X_l(p)X_{l+1}^*(p+v)}\right) + \angle\left(\frac{R_l^*(p+N_s-1)R_{l+1}(p+v)}{X_l^*(p+N_s-1)X_{l+1}(p+v)}\right)\right] \quad (69)$$

- Where
  - $p \rightarrow$ Frequency domain argument (Vertical axes in the grid).
  - $l \rightarrow$ Time domain argument. (Horizontal axes in the grid)
  - $R_l(p)R^*_{l+1}(p+v) \rightarrow$ Indicates to the two received consecutive OFDM symbols.
  - $X_l(p)X^*_{l+1}(p+v) \rightarrow$ Indicates to the two generated consecutive OFDM symbols.
  - $v \rightarrow$ Indicates to the v-shift.
  - $\tau_r \rightarrow$ Residual time offset in samples
  - $\varepsilon_r \rightarrow$ Residual frequency offset in samples.
  - $N \rightarrow$ number of elements in the single sub-frame
  - $N_{CP} \rightarrow$ Length of CP.
  - $K_1, K_2 \rightarrow$ Constants.

2.9.1.2. Residual Frequency Offset (RFO):

RFO is estimated by tracking the frequency offset effects on NRS signals of different OFDM symbols. The NRS signals positioned on the same subcarrier index of different OFDM symbols are considered for RFO estimation as they are uniformly affected by RTO. Four such pairs of NRS exists in each sub-frame of NB-IoT frame.

OFDM symbols which are Ns symbols apart carry NRS on same subcarrier indices. So, we estimate the RFO using conjugate product –as in equation [70] – between pair symbols one of them are located at l position and the other are in $l + N_s$, which are N_s symbols apart between both NRSs on the same sub-frame.

$$\frac{R_l\ (p)R^*_{l+N_s}(p)}{X_l\ (p)X_{l+N_S}(p)} = K_3 e^{\frac{-j2\pi N_s((N+N_{CP}))\varepsilon_r}{N}} \tag{70}$$

The phase of conjugate product on one pair of considered samples is proportional to the frequency offset as in equation [71]

$$\varepsilon_r = \frac{-N}{2\pi N_s(N+N_{CP})}\left[\angle\left(\sum\frac{R_l(p)R^*_{l+N_S}(p)}{X_l(p)X^*_{l+N_S}(p)}\right)\right] \tag{71}$$

2.10. P/S and NRS removal:

Starting from demodulation block to CRC block the inputs must enter serial and the input must contain the data only without NRS symbols as the transmitted data at the transmitter doesn't contain NRS and it is inserted in the middle of the transmitter chain. So, NRS symbol must be removed from the data get out from the equalizer and as demodulation must be serial it should also change the parallel input into serial output.

2.11. Demodulation

There are two main schemes of modulation in NB-IoT, QPSK and BPSK, both are supported in uplink and only QPSK is supported for downlink. QPSK is a modulation scheme maps every to bits to one of four symbols on the scheme constellation as shown in Figure 34. In this section we discuss the demodulation operation, which will de-map the received – after noise and channel effects – to '0' or '1' based on the below table. We faced two technique to operate demodulation, hard decision and soft decision demodulation using Log Likelihood Ratio (LLR). Soft decision gets BER better than hard decision, but due to using QPSK modulation which gives small BER compared to e.g. 16QAM we can use hard decision technique.

Table 3 QPSK constellation

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

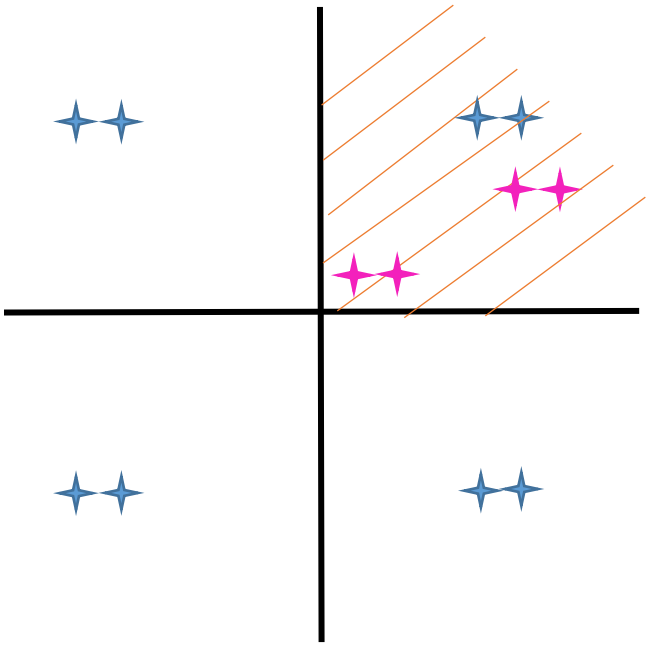


Figure 34 QPSK constellation

2.12. Descrambling:

Scrambling is an important block in communication systems which used to randomize the stream of data before doing any operation on it e.g. mapping. The objective of using it is to eliminate long run of zeros or ones to avoid both idle and reset cases and helps in clock recovery and having no DC component. Also, changing in the scrambling codes helps in security.

Scrambling / Descrambling implementation is based on pseudo random sequence that generated from 31-bit linear feedback shift register (LFSR) and XOR gates as shown in Figure 35. The two sequences of the two LFSR get XORed and the output is called a Gold sequence.

Descrambling Operation in NB-IoT

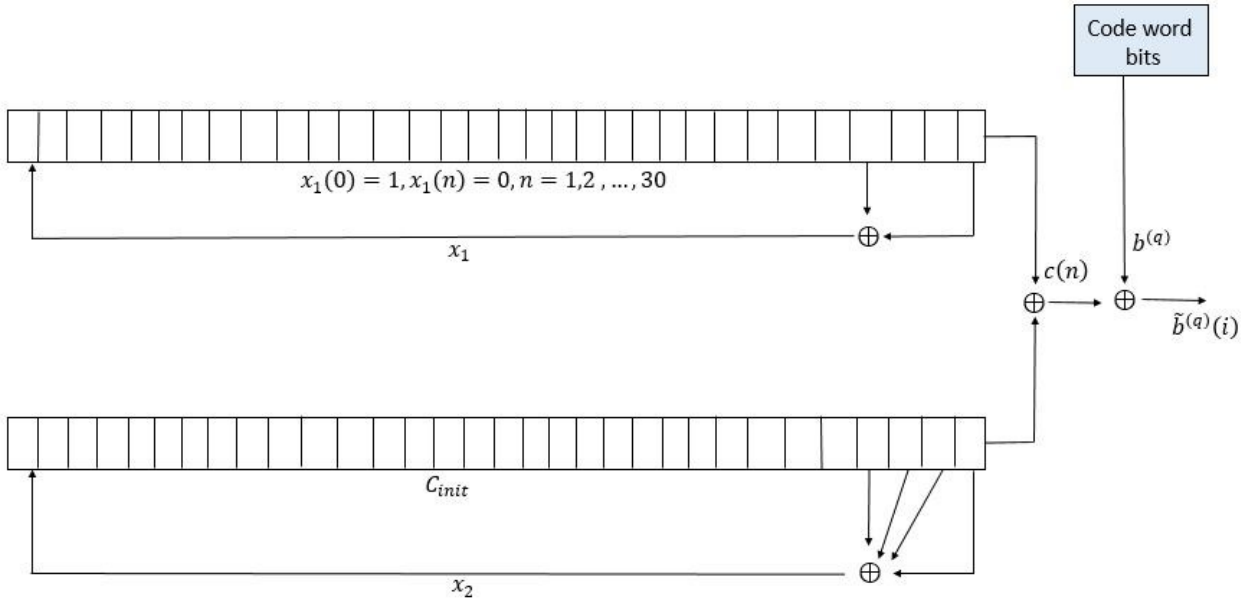


Figure 35 Descrambling Algorithm

The sequence of each LFSR is configure by the following two polynomials [72],[73]:

$$x_1(n) = 1 + D^3 + D^0 \quad (72)$$

$$x_2(n) = 1 + D^3 + D^2 + D^1 + D^0 \quad (73)$$

Then the two outputs get XORed to generate gold sequence c_n. Finally, the gold sequenced get XORed with the input data.

Descrambler initialization in NB-IoT:

LFSRs of the Gold sequence generator must be initialized,

x_1 will be initialized as follow:

$$x_1(0) = 1, x_1(n) = 0, n = 1,2, \ldots,30 \quad (74)$$

x_2 will be initialized using the following equation:

$$C_{init} = \left\{ n_{RNTI} . 2^{14} + q . 2^{13} + \left(\frac{n_s}{2}\right) . 2^9 + N_{ID}^{cell} \right\} \; for \; PDSCH \quad (75)$$

2.13. Rate De-Matcher [41:50]

Rate Matcher

The main function of the rate matcher block is improving the channel efficiency, which can be done by changing the code rate of the transmitted data. With the purpose of generating any arbitrary code rate, the rate matching procedure repeats, for code rates less than $\frac{1}{3}$, or punctures, for code rates greater than $\frac{1}{3}$, the bits of the mother code word [47].

The rate matcher consists of 3 sub-blocks which are called sub-blocks inter-leavers, these blocks take their inputs from Tail biting convolutional encoder, collect these bits in a virtual circular buffer whose size is 3 times that of one sub-block inter-leaver, then select or prune bits from the circular buffer to be transmitted [49].

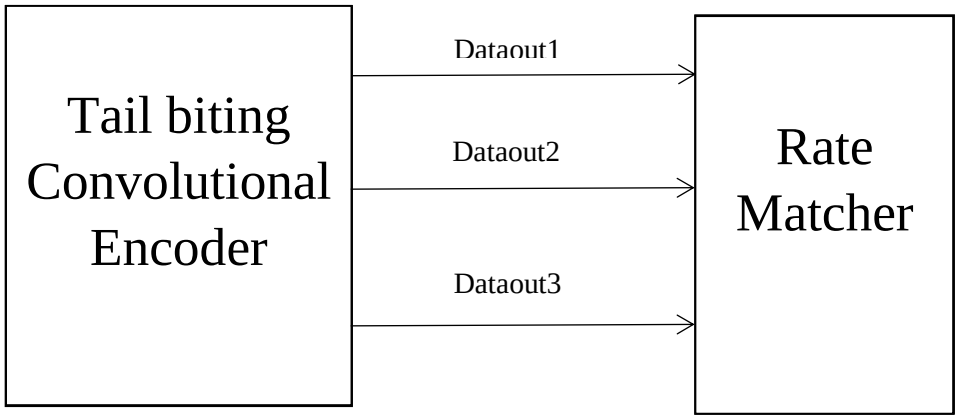
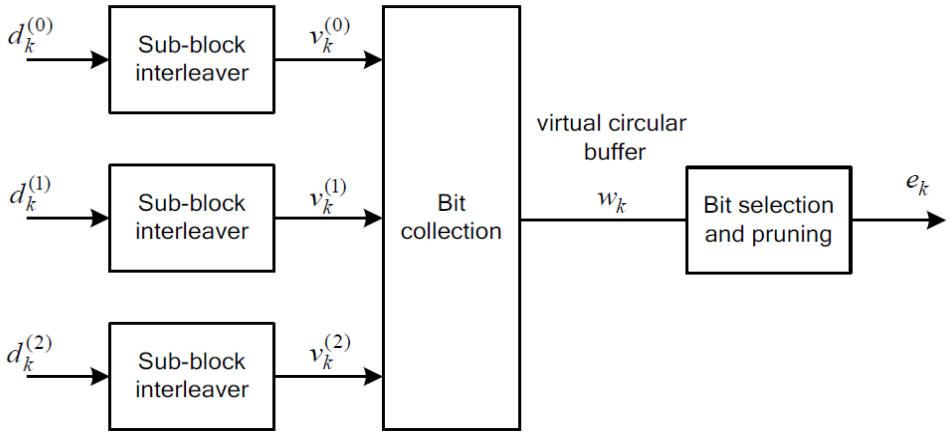


Figure 36 Rate Matcher in transmitter

2.13.1.1. Rate Matcher Algorithm



2.13.1.1.1. Sub-block inter-leaver

The inputs to sub block inter-leavers are [$d_k^{(1)}$, $d_k^{(2)}$, $d_k^{(3)}$] which are the outputs from the tail biting convolutional encoder block. Where "k" is varying from "0" to "D-1", where "D" is the number of bits. The following points explain the inter-leaver algorithm [49].

1- Assign the number of columns to be 32 columns in these sub-blocks which will be called $C_{subblock}^{CC}$, these columns will be numbered from "0" to"$C_{subblock}^{CC} - 1$".

2- Determine the number of rows in the matrix by finding minimum integer such $D \leq number\ of\ columns\ x\ number\ of\ rows$, where $number\ of\ rows$ will be $R_{subblock}^{CC}$, these rows will be numbered from "0" to "$R_{subblock}^{CC} - 1$".

3- If $(R_{subblock}^{CC}\ x\ C_{subblock}^{CC}) > D$, then $N_D = \left((R_{subblock}^{CC}\ x\ C_{subblock}^{CC}) - D\right)$ where N_D is the number of dummy bits. dummy bits are padded such that y_k = <NULL> for k = 0,1,2,…..,N_D-1, then $y_{N_D+k} = d_k^{(i)}$, k= 0,1,2,….,D-1. The bit sequence y_k is written into the matrix:

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

4- Perform the inter-column permutation for the matrix based on the pattern $P(j)_{j\in\{0,1,2,\ldots,C_{subblock}^{CC}\}}$ that is shown in the following table, where P(j) is the original column position of the j^{th} permuted column. After permutation of the columns, the inter-column permuted matrix $(R_{subblock}^{CC}\ x\ C_{subblock}^{CC})$ is equal to

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

Table 4 Permutation Table

Number of columns $C_{subblock}^{CC}$	**Inter-column permutation pattern** (P(0), P(1),……, 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>

5- The output of the block inter-leaver is the bit sequence read out column by column from the inter-column permuted $(R_{subblock}^{CC}\ x\ C_{subblock}^{CC})$ matrix. The bits after sub-block interleaving are denoted by $v_0^{(i)}, v_1^{(i)}, \ldots, v_{k_\pi-1}^{(i)}$. Where $v_0^{(i)}$ is corresponds to $y_{P(0)}$, $v_1^{(i)}$ is corresponds to $y_{P(0)}\ y_{P(0)+\ C_{subblock}^{CC}}$ and $k_\pi = (R_{subblock}^{CC}\ x\ C_{subblock}^{CC})$.

2.13.1.1.2. Bit collection, selection, and transmission

The circular buffer length is $k_w = 3k_{\pi}$ which is generated as follows [49]:

$w_k = v_k^{(0)} \; for \; k = 0,1, \dots, k_{\pi} - 1.$

$w_{k+k_{\pi}} = v_k^{(1)} \; for \; k = 0,1, \dots, k_{\pi} - 1.$

$w_{k+2k_{\pi}} = v_k^{(2)} \; for \; k = 0,1, \dots, k_{\pi} - 1.$

Denoting by E the rate matching output sequence length, the rate matching output bit sequence is e_k, k = 0, 1,..., E -1. Where "E" is upper layer parameter and input to the block.

Set k = 0 and j = 0: Each time it starts from the beginning of the buffer not as turbo.

while $\{k < E\}$: it iterates till reach the length of data which is required.

if $w_{j \; mod \; k_w} \neq < NULL >$: ignoring the dummy bits which was added in the interleaver.

$e_k = w_{j \; mod \; k_w}$: The output is the data inside the circular buffer.

k = k + 1.

end if

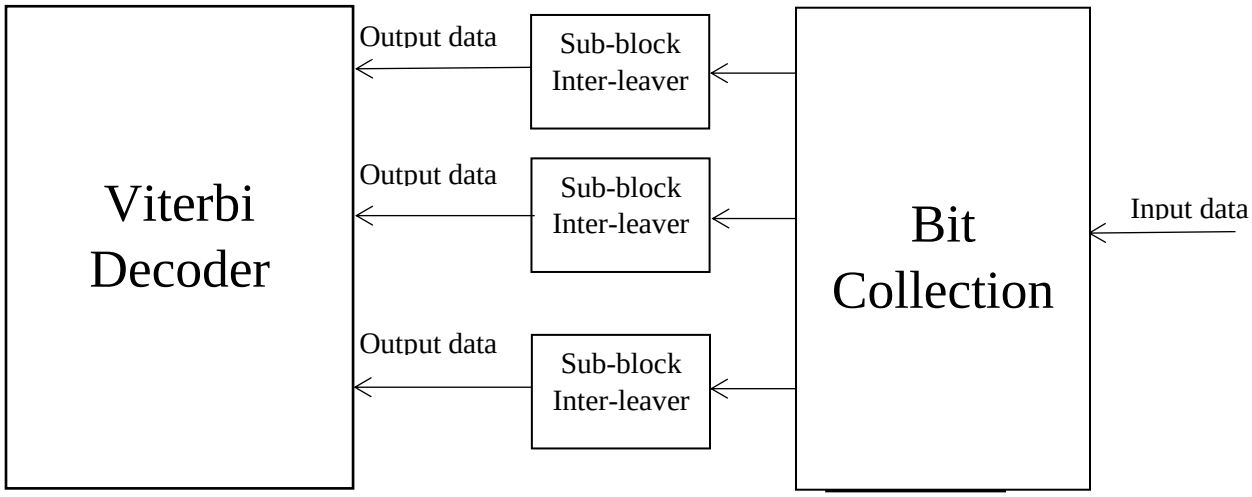
j = j + 1.

end while

Rate De-Matcher:

The function of this block is to reverse the rate matcher process, the output of this block is three original data streams that were generated from the tail biting convolutional encoder in the transmitter [43].

2.13.1.2. Rate De-Matcher Algorithm



2.13.1.2.1. Bit Collection

This block consists of one memory with size of (3 x (max TBS +24) * log_2(max repetition size)) which will be (7680 * 12), and a control unit to manage the data stream saving in this memory in all cases. The size of this memory is large, because in this release the max TBS is "2536 bits" [51] as mentioned in the standard, and the maximum repetitions is "2048 repetitions".

The input to this block is the bit stream, which is the circular buffer data stream in the transmitter with any length according to the upper layer parameter "E". There are three cases to deal with this data:

1- If the data length is as circular buffer length: this will be saved as it and then will be passed to the inter-leavers to be permutated.

2- If the data length is greater than the circular buffer length which occurs due to the repetition of data in the circular buffer: this will be saved to reach the circular buffer length then the remained data will be added to they until they end, then get the average of these data.
3- If the data length is less than the circular buffer length which occurs due to the puncturing of data in the circular: in this case the data will enter the buffer till its length, then padding zeros to it till reaches the circular buffer length.

2.13.1.2.2. Sub block de-inter-leaver

This block consists of four memories each one is (number of columns * number of rows for the max TBS$\left(\frac{\text{Max } TBS+24}{32}\right)$) which is (32*80) [49]. This block should be 3 memories to store the whole data, but here there are four memories, the fourth memory is used for the consecutive input stream. This means if the data stream is still inside the three rams and new input come to the block it will be saved in the first ram, and the remained three memories will pass the data to the next block at the same time. This block deals with the input as the transmitter deals.

2.14. Viterbi Decoder

In Communication Systems, when data represented in bits is sent from source to destination, they suffer from corruption due to the presence of noisy channels, Channel coding techniques is used to overcome the effect of those noisy channels and interference and ensure that the received information is as close as possible to the transmitted information. Which results in improve in the BER and improve reliability of information transmission.

Forward error correction (FEC) is one of the techniques used to detect and correct errors in the transmitted data without the need for retransmission. Convolution Encoding with Viterbi Decoding is a powerful method for Forward Error correction and Detection, where data is convoluted and then transmitted into a noisy channel, this process involves the insertion of redundant data which helps in detecting and correcting errors due to those noisy channels. Decoding of convoluted data is done using Viterbi algorithm which appears to be the most powerful method in terms of efficiency and bit error rates. The algorithm tracks down the most likely sequences the encoder went through in encoding the data, and uses this information to discover the original message.

Convolutional Codes

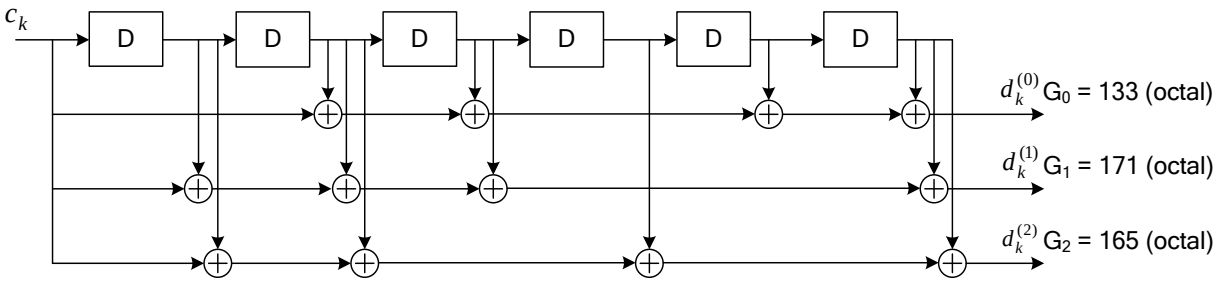


Figure 37: Rate 1/3 tail biting convolutional encoder

Convolutional encoder is implemented using shift registers. The contents of the shift register are known as the state of the encoder. At each time instant contents of the shift register are shifted declaring a new state for the encoder, The bit sequence input for a given code block to channel coding is denoted by $c_0, c_1, c_2, c_3, ..., c_{K-1}$, where K is the number of bits to encode. After encoding the bits are denoted by $d_0^{(i)}, d_1^{(i)}, d_2^{(i)}, d_3^{(i)}, ..., d_{D-1}^{(i)}$, where D is the number of encoded bits per output stream and i indexes the encoder output stream. The relation between and between K and D is dependent on the channel coding scheme.

A (k, n, m) can be seen as a convolutional encoder having m stages, k bits entering the shift register at each time instant and n output bits. Output bits are formed using generator sequences:

$$G_0 = [1\ 0\ 1\ 1\ 0\ 1\ 1], \qquad G_0 = 1 + 2 + 8 + 16 = 133(octal)$$

$$G_1 = [1\ 1\ 1\ 1\ 0\ 0\ 1], \qquad G_1 = 1 + 8 + 16 + 32 + 64 = 171(octal)$$

$$G_2 = [1\ 1\ 1\ 0\ 1\ 0\ 1], \qquad G_2 = 1 + 4 + 16 + 32 + 64 = 165(octal)$$

The rate of the encoder is denoted by $R = k/n$.

Tail Biting Convolutional Coding

Tail biting technique overcomes the problem of the added cost of transmitting extra termination bits experienced by trellis terminations. A tail biting convolutional code with constraint length 7 and coding rate 1/3 is defined in Figure 37. The initial value of the shift register of the encoder shall be set to the values corresponding to the last 6 information bits in the input stream so that the initial and final states of the shift register are the same. Therefore, denoting the shift register of the encoder by $s_0, s_1, s_2, ..., s_5$, then the initial value of the shift register shall be set to

$$s_i = c_{(K-1-i)}$$

The encoder output streams $d_k^{(0)}$, $d_k^{(1)}$ and $d_k^{(2)}$ correspond to the first, second and third parity streams, respectively as shown in Figure 37.

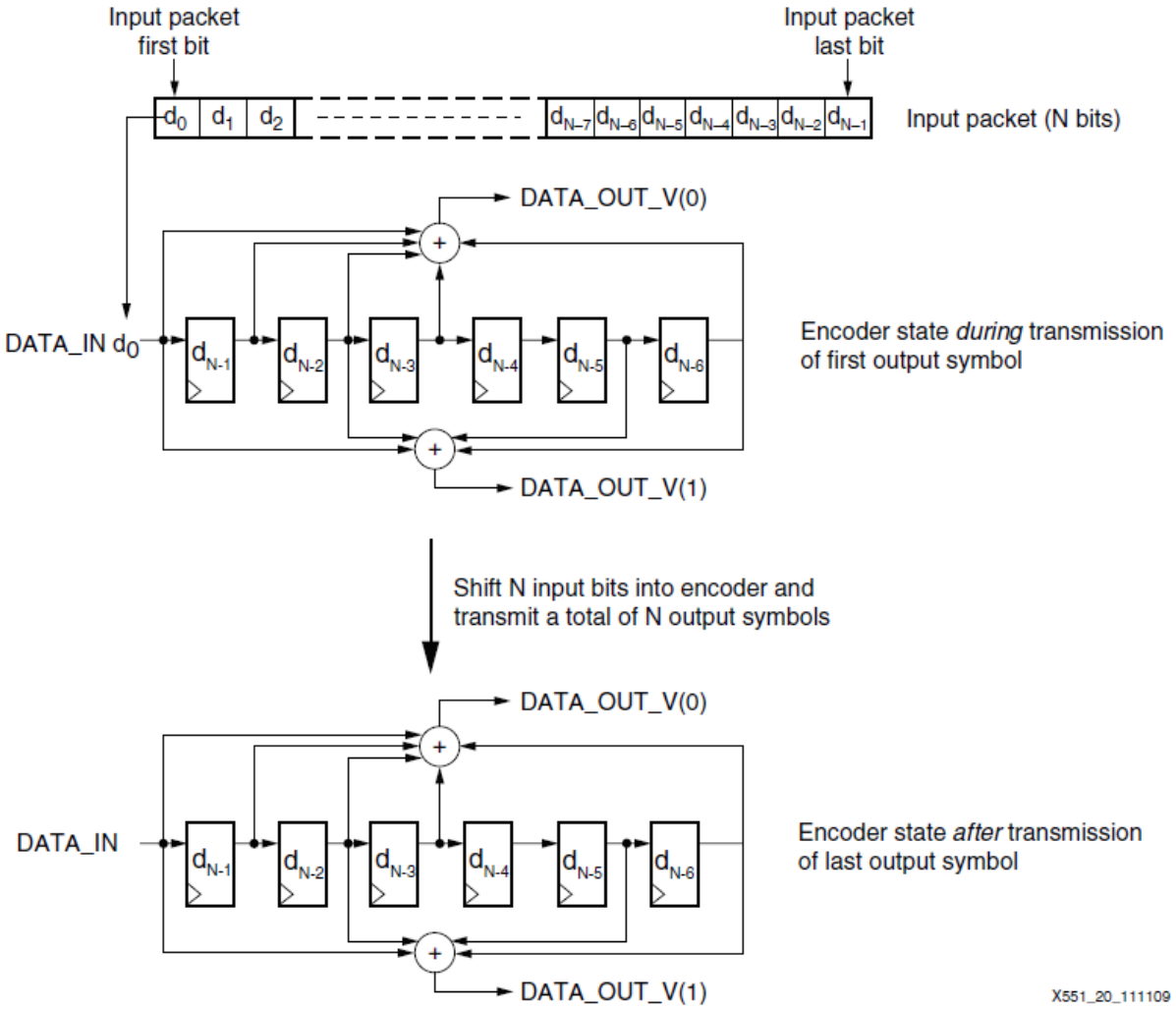


Figure 38: Convolutional Encoding with tail bits

Tail biting is done by:

1) The encoder shift register is first initialized with the last n data bits before packet transmission where $n = Constraint\ length - 1$, Resulting in an equal initial and final state.
2) For constraint length=7, the encoder is initialized with the last six bits of the packet, Transmission starts when Data=d0 and last symbol is formed when Data=Dn-1. One more shift puts the encoder back into the initial state where decoder can begin tracing back the packet from that state.

Encoding with this technique has the advantage of:

1) Code rate is not affected.
2) Error correction properties of the convolution code are not affected.

But also having disadvantages of:

1) Decoding latency is increased due to the fact that initial and final states are required to correctly start tracing back.
2) Receiver complexity is increased.

Viterbi Decoder Description

Viterbi decoder is a maximum likelihood method to detect the most probable sequence of hidden states. Viterbi decoder uses a tree search procedure to optimally detect the received sequence of data. It calculates a measure of similarity between the received signal and all the trellis paths entering each state at time. Remove all the candidates that are not possible based on the maximum likelihood choice. Remove all the candidates that are not possible based on the maximum likelihood choice. When two paths enter the same state, the one with the best path metrics (the sum of the distance of all branches) along the path is chosen. This path is called the surviving path. This selection of surviving paths is done for all the states and makes decisions to eliminate some of the least likely paths in early calculation stages to reduce the decoding complexity. A more redundant description of the same process is shown in Fig. 39, this description is called trellis.

2.14.1.1. Trellis Diagram

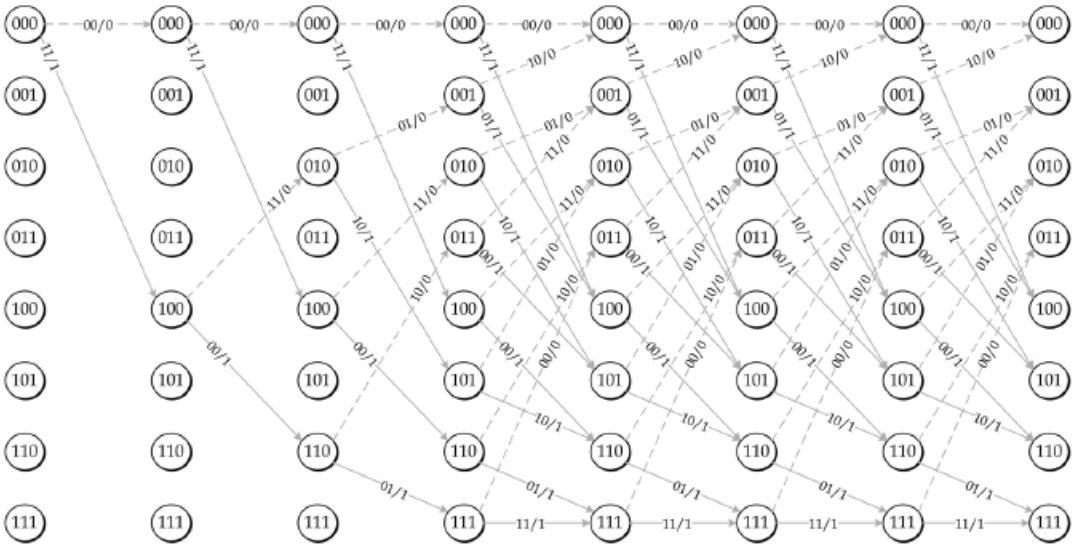


Figure 39: Trellis Transitions for Constraint length=4

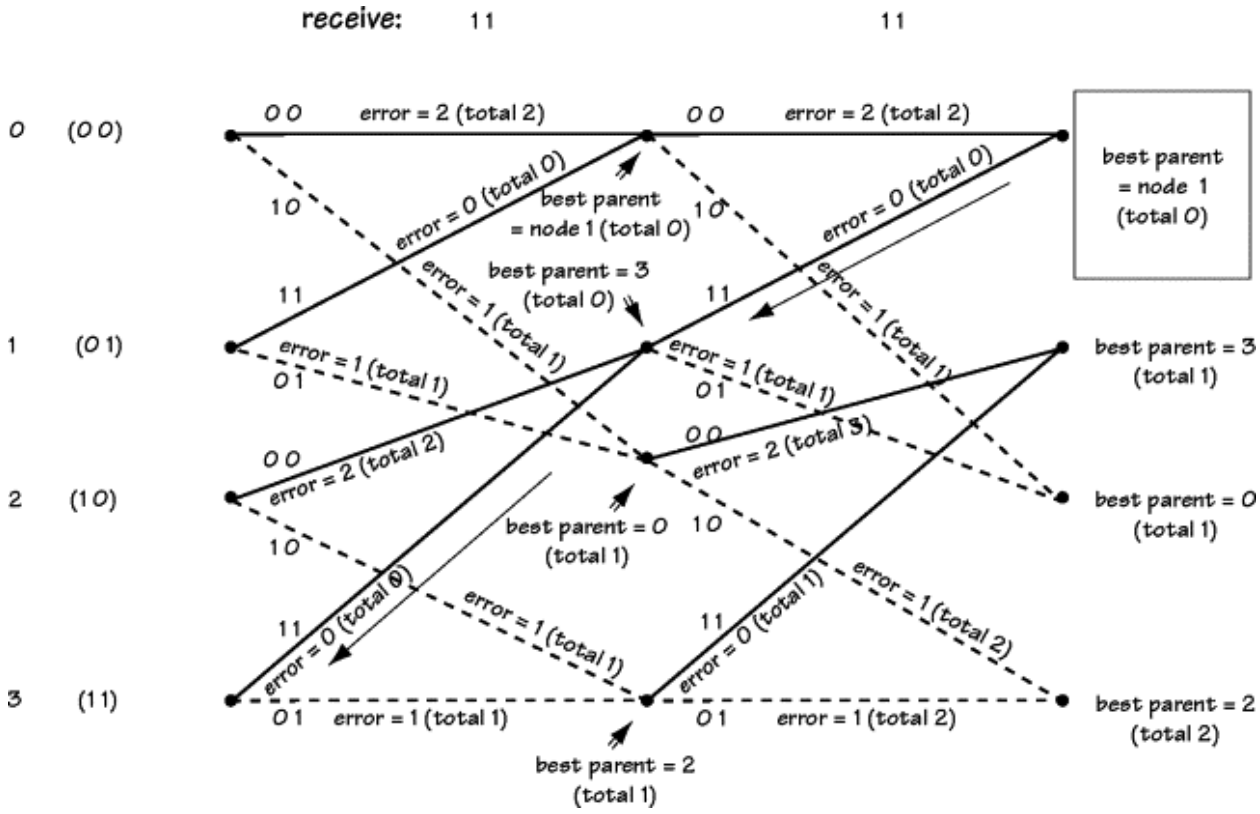


Figure 40: Error computation for constraints length=3

Trellis Diagram states is decided according to the constraint length. Consider K and being the constraint length thus trellis states would be equal to 2^{k-1}. For each received sequence of bits several computations take place. Those computations decide which path is best suited or the path which faced the least number of errors based on trellis expected outputs. In Figure 39, Trellis is implemented for constraint length equal to 4 and rate equal to $1/2$, thus having 2^3 states. Each received sequence goes through a no. of computations starting with Branch Metric computation. Where Hamming Distance is calculated. Hamming Distance is the measure of how likely is the received sequence to the Expected outputs. Followed by path metric computation where errors are accumulated and saved to be used in decision making. At the end of the received sequence, trace back procedure start from the path with the least Number of errors.

Hidden states or observed sequence is obtained using the convolutional encoder. The computed sequence is based on state machine pattern as shown in Figure 40. The encoder tells the receiver about its movement through the state machine. Viterbi Algorithm operates on that state machine assumption.

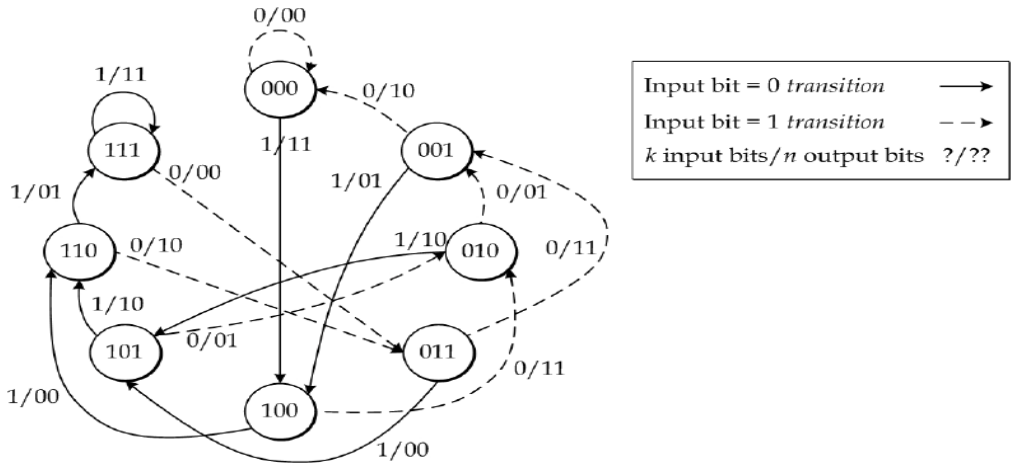


Figure 41: State Diagram for constraints length=4.

State diagram shows 8 states with all possible transitions in case of the input bit is 0 or 1. To better understand how Viterbi traces the movement through the state machine consider the example shown in Figure 42 and Figure 43. For the sequence (1011000), the hidden states obtained according to the state diagram starting from state 000 are (11 11 10 10 10 11 10).

Case 1:

In case of no error occurred, the received sequence is (11 11 10 10 10 11 10). The decoder starts from State 000, followed by a sequence of steps which could be summed up into:

1) For each state there is two possible branches, one for the case of the input being 1 and one for the input being 0, for each case there is a possible output as shown in Figure. For the received sequence 11 starting from state 000, the branch output which matches the received sequence is the second branch. Same step is repeated for each received sequence until the end of message.
2) The survived path is now computed so trace back procedure start from state 000 at t8, Extracting an output of (0001101). This result is saved in a LIFO memory.

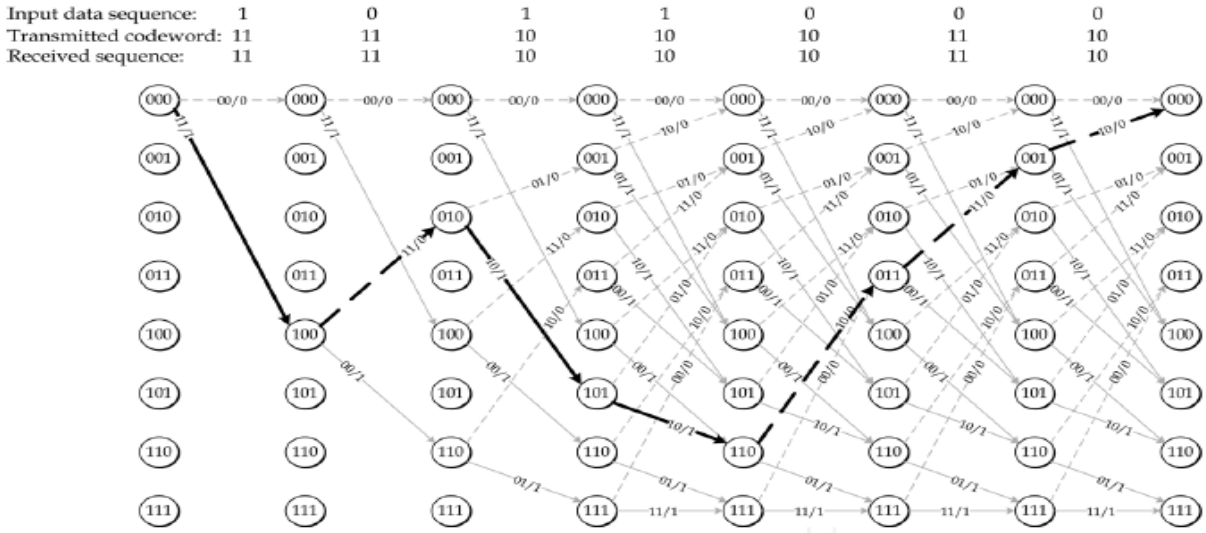


Figure 42: Computed Distances for different paths

Case 2:

Consider the received sequence being *10111010101110,* thus error occurred in second bit so corruption occurred due to the presence of noisy channels, the procedure that the Viterbi go through is:

1) Starting from state 000, the received sequence suffers an error =1 in both paths, so it is up to the algorithm chosen to choose whether to continue with the first path or the second path, if number of errors in one of the paths is smaller than the other, this path is chosen and the other one is discarded, then same procedure is taken for each bit stream till the end of message.
2) After computing the survival path, Trace back unit computes the decoded sequence as before.

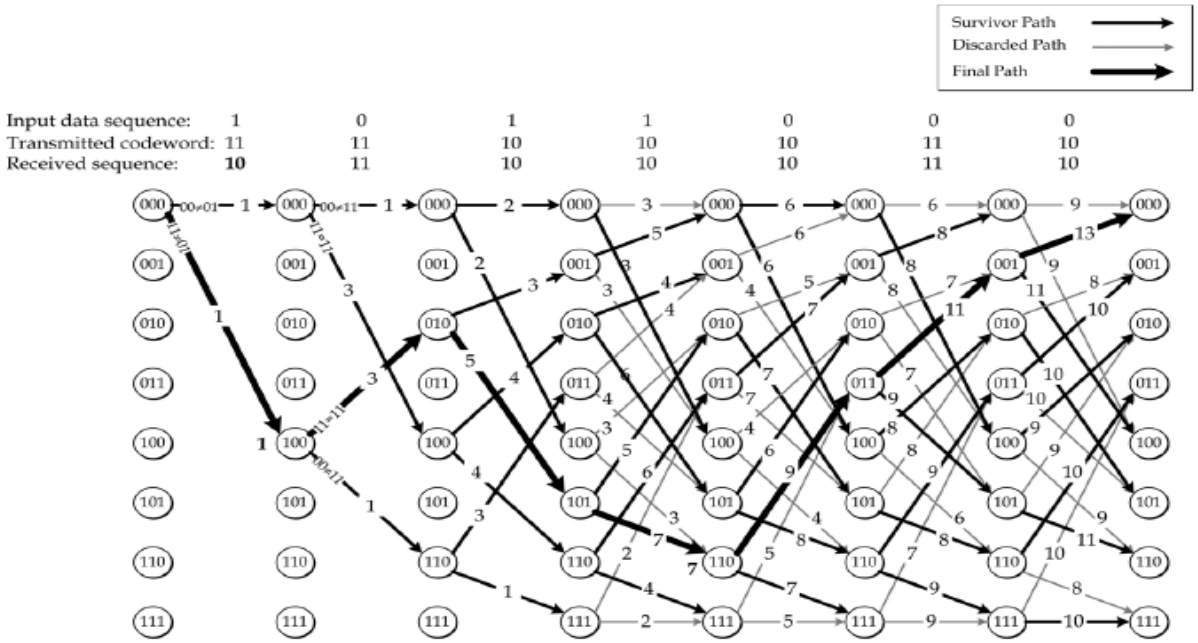


Figure 43: Survivor path computation

1.14.4. Decoding of Convolutional Codes

2.14.1.2. 1.14.4.1. Two Step SOVA-Based Decoding Algorithm for Tail biting Codes.

In this method the Viterbi decoding is done on two fixed steps one for computing most likely paths and the other one is to start decoding from a specific state based on the information taken from the previous step

Decoding steps:

1. Start from all initial states with path metrics set to zero.
2. Compute errors throughout the trellis stages.
3. Choose the final state with the minimum accumulated metric.
4. Trace back the chosen survival path and record likelihood of the states.
5. Based on the previous step an initial and final state is chosen and a second Viterbi is done.

Disadvantages:

1. Large number of processing steps.
2. An error could occur in case of a bad selection of initial and final states.

2.14.1.3. 1.14.4.2. Modified Circular Viterbi Algorithm.

The chosen algorithm is based on performing several iterations of the Viterbi algorithm until a tail biting pattern is found,

Decoding Steps:

1. Start the Viterbi algorithm with initial metrics set to zero.
2. Compute the survival path based on the path with the least accumulated metric.

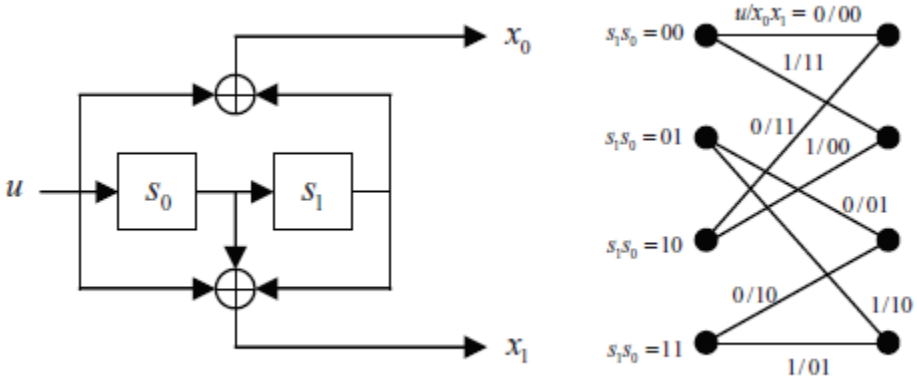


Figure 44: Trellis Transitions

3. If this path is a tail biting path, decoding stops and the survival path is decoded.

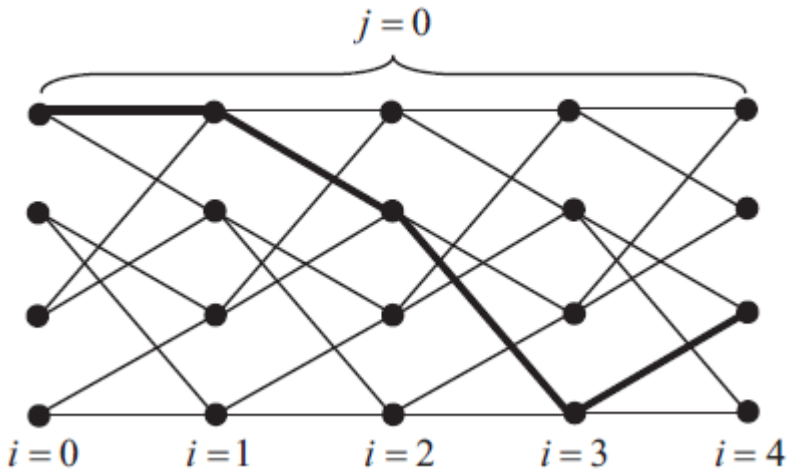


Figure 45: Trellis Transition at the end of the first iteration.

4. If the survival path is not a tail biting path, metrics are updated with final metrics of the previous iteration.
5. Apply the Viterbi algorithm on the same message, until a new survivor path is obtained.
6. If the new survivor path is a tail biting path, algorithms stops.
7. If maximum number of iterations is reached, algorithm stops and the best survival path is decoded.

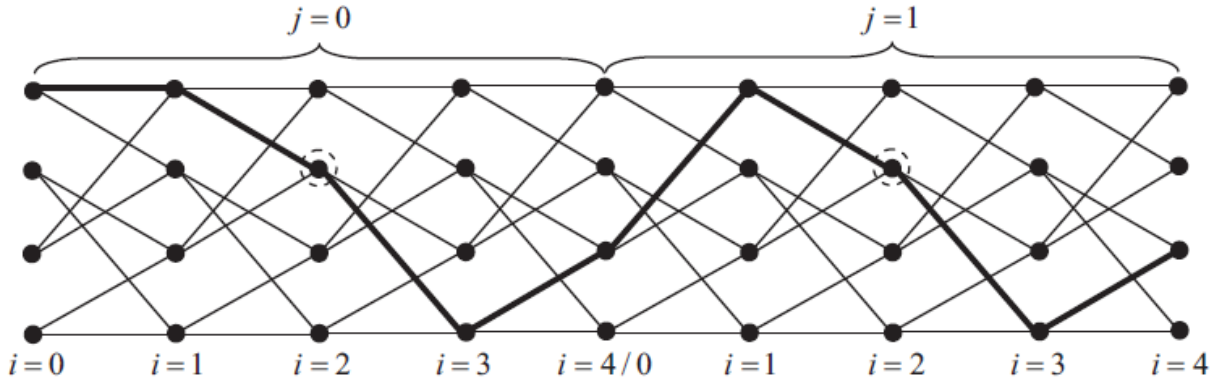


Figure 46: Trellis at the end of the second iteration.

2.15. Cyclic Redundancy Check

Cyclic Redundancy check (CRC) is an error detecting code. It is used commonly in digital networks to detect any changes in the row data. In the transmitter it appends bits to the row data which is the remainder of a polynomial division, and in the receiver, it uses the same technique as the transmitter if the remainder was zero that means the data is correct, otherwise the data is wrong.

Cyclic Redundancy Check Algorithm

Denote the input bits to the CRC computation by $a_0, a_1, \dots, a_{A-1}$, and the parity bits by $p_0, p_1, \dots, p_{L-1}$, where A is the size of the input sequence and L is the number of parity bits. The parity bits are generated by the following polynomial [51].

$$crc24(D) = D^{24} + D^{23} + D^{18} + D^{17} + D^{14} + D^{11} + D^{10} + D^{7} + D^{6} + D^{5} + D^{4} + D^{3} + D + 1 \quad (76)$$

The bits after CRC attachment are denoted by $b_0, b_1, \dots, b_{B-1}$, where B = A+ L. The relation between a_k and b_k is:

$$b_k = a_k, \quad \text{For k = 0,1,2,…, A-1} \quad (77)$$

$$b_k = p_{k-A}, \quad \text{For k = A , A+1 ,…, A+L-1.} \quad (78)$$

There is no segmentation in NB-IoT, because the maximum TBS is 2536 and which is less than 6144 (the minimum number to segment the Transport block into code blocks).

2.16. RAM Sharing

Due to the large similarly-sized memory requirements in the synchronization block and the rate de-matcher block also the fact that the two blocks never simultaneously access their memories, it became appealing to share the ram resources of these blocks. Mainly, the sharing was made possible by multiplexing the two clocks of the blocks with the selection of the LOCKED signal which indicates the activity of the synchronization process, this signal was also used to multiplex the inputs to the ram, the pointers, and the enables. Also, a multiplexing process was implemented to help implement the rate de-matcher ram as multiple smaller rams. The ram module consists mainly of 15 separate rams, 14 of these are numbered in pairs real and imaginary from 1 to 7 and another single ram module annotated by ram0. The size of each ram matches its address width.

2.17. Design Specifications:

Table 5 Design Specifications

Parameter	Specs
Channel Model	ETA with a maximum Doppler shift of 5 Hz, $N_{Ch} = 9\ taps$, and maximum excess tap delay of 5 us
Minimum Required SNR	$-12.6\ dB$
Frequency Offset	[-35:35] KHz, estimated with 50 Hz accuracy
Time Offset	[0:19200] samples, estimated with a maximum error of 2 samples
CORDIC stages	15
FFT size	16
Decoding	Hard
Modulation scheme	$QBSK$
System BW	$200\ KHz$
Sampling frequency	$1.92\ MHz$

- Output data rate for NPDSCH layer:
  - Output clock: clk_260 = 3.84 Mbps
  - Calculated max peak Rate according to our design:
    - Max TBS = 2536 including pilots, and coding 1/3 then take 24 sub-frame including NPSS, NSSS overhead, and pilots,

$$\text{Max Rate} = \frac{2536}{24ms + 6ms} = 84.533\ Kbps$$

    - Max TBS = 2536 including pilots, and coding 1/1 then take 8 sub-frames including NPSS, NSSS overhead, and pilots,

$$\text{Max Rate} = \frac{2536}{8ms + 2ms} = 253.6\ Kbps$$

  - Calculated for any NPDSCH layer

$$\text{Max Rate} = \frac{336}{1ms} = 336\ Kbps$$

- Calculated rates according to some assumptions:
  - our TBS = 296 including pilots, and coding 1/3 then take 3 sub frames with respect to upper layer

$$\text{Max Rate} = \frac{296}{3\text{ms}} = 98.666\ \text{Kbps}$$

- Whole Chain Latency with TBS = 296:
  - Clock cycles: 2793 clk_520.
  - Latency delay time: 1.45 ms.
- Clocks used: clk_520 = 520 ns, clk_260 = 260 ns, clk_130 = 130 ns

Chapter 3
Design implementation

3.1. Coarse Synchronizer

As the block implements its functionality as a finite state machine, no detailed block diagrams are presented in this part. There is, however, a state diagram explaining the block as a state machine and its operation and naturally, there is a top-level block diagram illustrated in the following figure and the inputs and outputs of this block diagram are defined in Table 8 and Table 9. Firstly, in Table 6 some statistics are illustrated about this implementation and its use of important resources.

Table 6 Usage of Important Components and Resources:

Resource	Usage
Ram	10.3KB
Complex Multipliers	Real Multiplications: 2 of (1Byte*1Byte)
	Real Multiplications: 2 of (4Bits*1Byte)
Real Multipliers	2 of (2Bytes*2Bytes)
	2 of (28Bits*28Bits)
	1 of (28Bits*17Bits)
CORDIC Units	1 arctan-calculator unit
	5 phase rotator units

3.1.1. Block Diagram

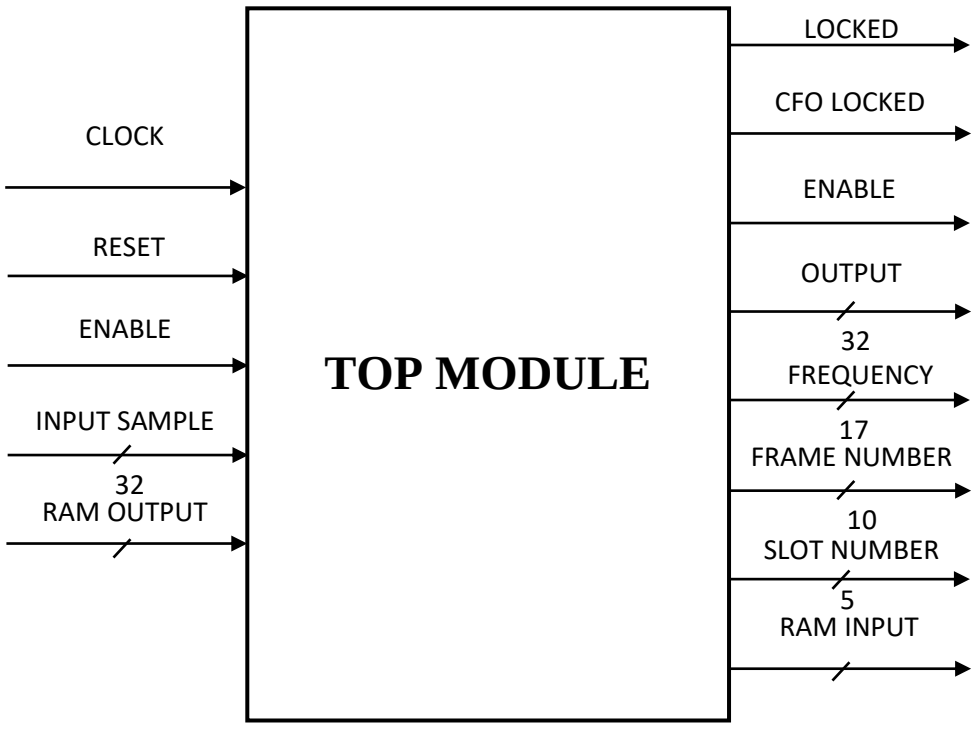


Figure 47 CSYNCH Top-Level Block Diagram

3.1.2. State Diagram

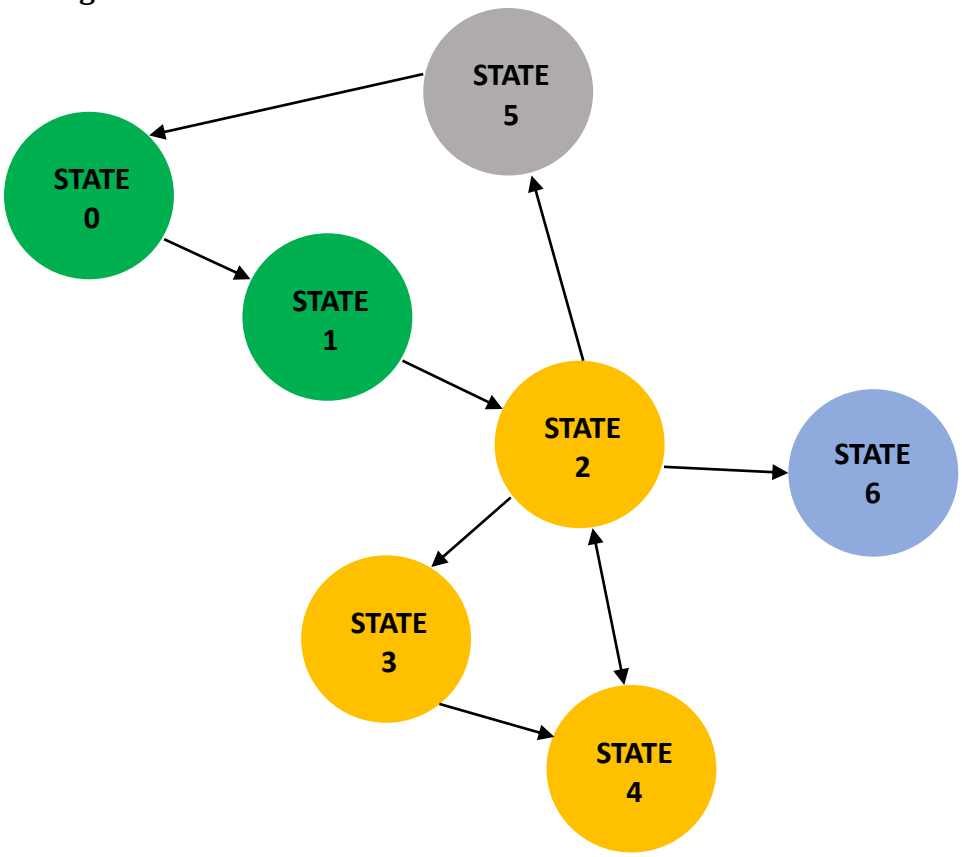


Figure 48 Simplified State Diagram of the Block

Let the states illustrated in Figure 48 Simplified State Diagram of the Blockbe defined as follows:

State 0: this state sets the stage for step 1 in the algorithm; it calculates the metric in the first frame and stores the metric data in a shared ram. It also implements the function of applying the code cover to the products.

State 1: this is mainly the state where step 1 of the algorithm is implemented as it updates the metric by averaging, it sorts the metric, and checks if the maximum passes the threshold.

State 2: this state serves mainly two purposes, it captures time-domain data required to implement the matched filter step and it also sorts and decides on estimates in step 2 and 3.

State 3: this state loads the time-domain samples of a complete NPSS symbol from a 65-element ROM.

State 4: this state calculates and updates the matched filter metric through the use of combinational CORDIC units.

State 5: this state prepares for going back to step 1 again in the case of a false auto-correlation peak.

State 6: this is the state of normal post-synchronization operation; it implements the functions of CP removal, down-sampling, and symbol-position data acquirement.

Table 7 Illustration of Transitions between States:

Current State	Next State	Event
S_0	S_1	Always after it is executed
S_1	S_2	When the maximum metric value crosses the assigned threshold
S_2	S_3	At the first time a matched filter step is executed
S_2	S_4	Every time a matched filter step is executed except for the first time
S_2	S_5	If the maximum of step 2 metric does not pass the assigned threshold
S_2	S_6	After all steps of the procedure are executed
S_3	S_4	Always after it is executed
S_4	S_3	Always after it is executed
S_5	S_0	Always after it is executed
S_6	…	Does not end

3.1.3. Interface Tables

As the used ram is a shared ram, it is outside the block and hence has an interface with the block. And because the used ram is accessed as separate 15 rams, it has its own interface table, Table 9, separate from the main interface table, Table 8.

Table 8 csynch Interfacing Table of the Top-Level Module:

Signal Name	Direction	Width	Description
Clock	Input	1	A 1.92MHZ clock signal
Reset	Input	1	An edge-sensitive negative logic global reset signal
Enable	Input	1	A level-sensitive signal that permits the block to operate
Input Sample	Input	32	The serial input samples from the ADC with 2Bytes for real part and 2Bytes for imaginary part
Ram Output	Input	…	Clearly specified in table 3
Locked	Output	1	A level sensitive signal that indicates the success and end of the synchronization procedure
CFO Locked	Output	1	A level sensitive signal that indicates the beginning of data receiving and it is used to enable the next block in the chain
Enable	Output	1	A level sensitive signal that informs the next block in the chain of a new valid sample
Output Sample	Output	32	The serial output samples of CP-removed down-sampled symbols
Frequency Estimate	Output	17	The final estimates of FFO and IFO added together as a frequency offset estimate
Frame Number	Output	10	A counter that indicates the number of frames processed since the block was enabled and it resets after 1024 frames
Slot Number	Output	5	A counter that indicates the number of slots processed in the current frame and it resets after counting 20 slots
Ram Input	Output	…	Clearly specified in table 3

Table 9 Block's Interfacing Table with Shared-Ram:

Signal Name	Direction	Width	Description
Ram0 Enable	Output	1	Ram0 write enable level-sensitive signal
Real Ram1 Enable	Output	1	Real ram1 write enable level-sensitive signal
Imaginary Ram1 Enable	Output	1	Imaginary ram1 write enable level-sensitive signal

Real Ram2 Enable	Output	1	Real ram2 write enable level-sensitive signal
Imaginary Ram2 Enable	Output	1	Imaginary ram2 write enable level-sensitive signal
Real Ram3 Enable	Output	1	Real ram3 write enable level-sensitive signal
Imaginary Ram3 Enable	Output	1	Imaginary ram3 write enable level-sensitive signal
Ram4 Enable	Output	1	Real&Imaginary ram4 write enable level-sensitive signal
Ram5 Enable	Output	1	Real&Imaginary ram5 write enable level-sensitive signal
Real Ram6 Enable	Output	1	Real ram6 enable level-sensitive signal
Imaginary Ram6 Enable	Output	1	Imaginary ram6 write enable level-sensitive signal
Ram7 Enable	Output	1	Real&Imaginary ram7 write enable level-sensitive signal
Ram0 Address	Output	8	Ram0 read&write address
Ram1 Address	Output	9	Real&Imaginary ram1 read&write address
Ram2 Address	Output	9	Real&Imaginary ram2 read&write address
Ram3 Address	Output	8	Real&Imaginary ram3 read&write address
Ram4 Address	Output	8	Real&Imaginary ram4 read&write address
Ram5 Address	Output	8	Real&Imaginary ram5 read&write address
Ram6 Address	Output	9	Real&Imaginary ram6 read&write address
Write Ram7 Address	Output	11	Real&Imaginary ram7 write address
Read Ram7 Address	Output	11	Real&Imaginary ram7 read address
Ram0 Write Data	Output	16	Ram0 input data
Real Ram1 Write Data	Output	16	Real ram1 input data
Imaginary Ram1 Write Data	Output	16	Imaginary ram1 input data
Real Ram2 Write Data	Output	16	Real ram2 input data
Imaginary Ram2 Write Data	Output	16	Imaginary ram2 input data
Real Ram3 Write Data	Output	16	Real ram3 input data
Imaginary Ram3 Write Data	Output	16	Imaginary ram3 input data
Real Ram4 Write Data	Output	16	Real ram4 input data
Imaginary Ram4 Write Data	Output	16	Imaginary ram4 input data
Real Ram5 Write Data	Output	16	Real ram5 input data
Imaginary Ram5 Write Data	Output	16	Imaginary ram5 input data
Real Ram6 Write Data	Output	16	Real ram6 input data
Imaginary Ram6 Write Data	Output	16	Imaginary ram6 input data
Real Ram7 Write Data	Output	16	Real ram7 input data
Imaginary Ram7 Write Data	Output	16	Imaginary ram7 input data
Ram0 Read Data	Input	16	Ram0 output data
Real Ram1 Read Data	Input	16	Real ram1 output data
Imaginary Ram1 Read Data	Input	16	Imaginary ram1 output data
Real Ram2 Read Data	Input	16	Real ram2 output data
Imaginary Ram2 Read Data	Input	16	Imaginary ram2 output data
Real Ram3 Read Data	Input	16	Real ram3 output data
Imaginary Ram3 Read Data	Input	16	Imaginary ram3 output data
Real Ram4 Read Data	Input	16	Real ram4 output data
Imaginary Ram4 Read Data	Input	16	Imaginary ram4 output data
Real Ram5 Read Data	Input	16	Real ram5 output data
Imaginary Ram5 Read Data	Input	16	Imaginary ram5 output data
Real Ram6 Read Data	Input	16	Real ram6 output data
Imaginary Ram6 Read Data	Input	16	Imaginary ram6 output data
Real Ram7 Read Data	Input	16	Real ram7 output data
Imaginary Ram7 Read Data	Input	16	Imaginary ram7 output data

3.1.4. Results

3.1.4.1. RTL Results

The following results were obtained twice from RTL simulations on a 100 random cases in the operating conditions mentioned earlier with randomized frequency and time offsets, assuming in one time in-band or guard-band deployment and assuming in another time standalone deployment, each under the same channel model mentioned earlier and assuming in each case the minimum SNR namely, -13dB and -5dB. Table 9 summarizes the results shown in figures 49 to 54. The residual frequency offset estimates are well below 0.05 of the carriers spacing which according to [53] results in an SNR degradation below 0.5dB at the minimum SNRs. The residual time offset and latency results are comparable to those in [7] and [11].

Table 10 csynch RTL Simulation Results Summary:

Metric	Standalone Deployment	In-Band or Guard-Band Deployment
PSS-Detection Latency (50% percentile)	$90ms$	$200ms$
PSS-Detection Latency (90% percentile)	$140ms$	$370ms$
PSS-Detection Latency (95% percentile)	$170ms$	$480ms$
False Alarm Probability	0%	1%
Timing Error	$-1.04\mu s \sim 1.04\mu s$	$-1.04\mu s \sim 1.04\mu s$
Frequency Estimation Error (95% percentile)	$-30Hz \sim 30Hz$	$-60Hz \sim 60Hz$

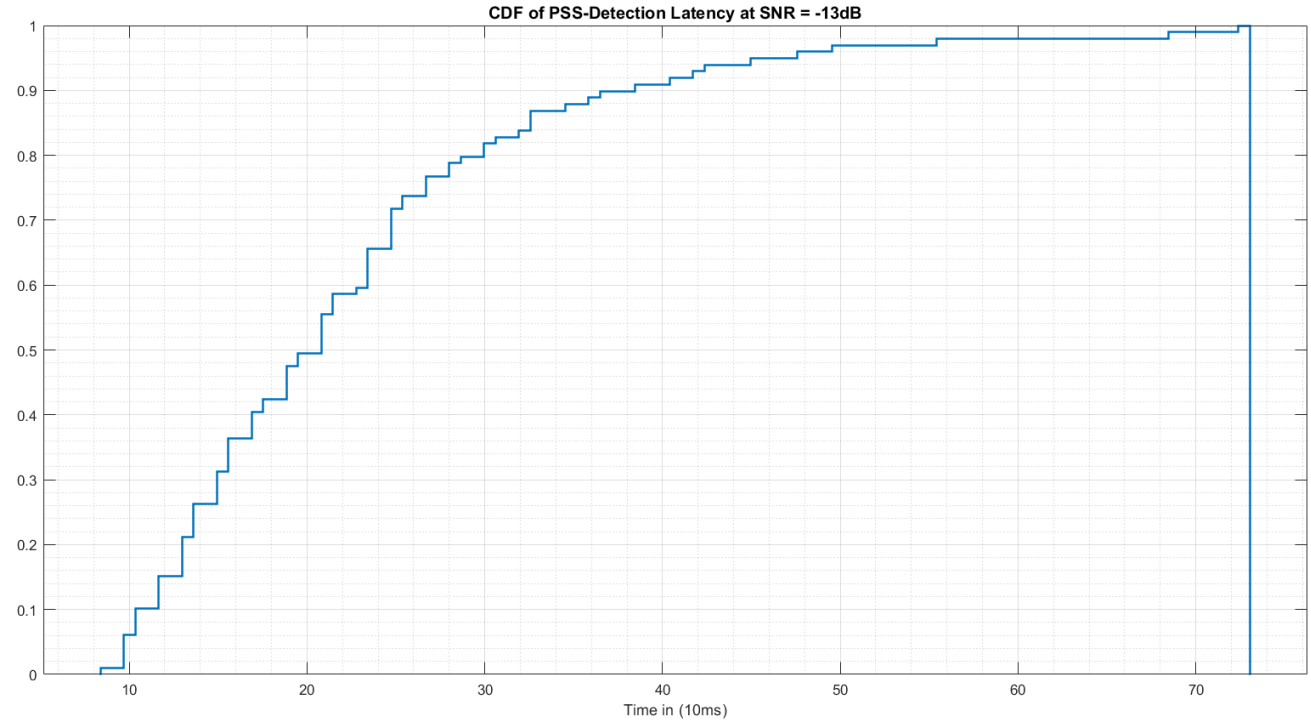


Figure 49 CDF of PSS-Detection Latency in In-Band or Guard-Band Deployment

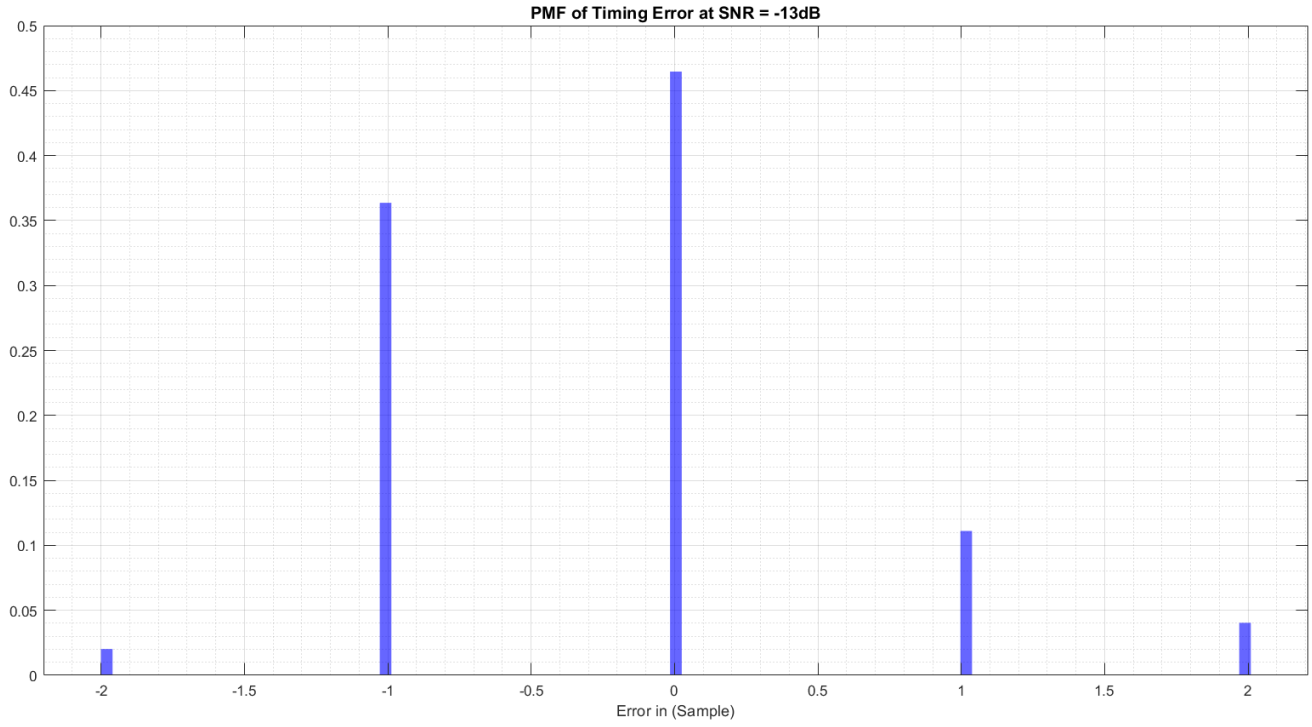


Figure 50 PMF of Synchronization Timing Error in In-Band or Guard-Band Deployment

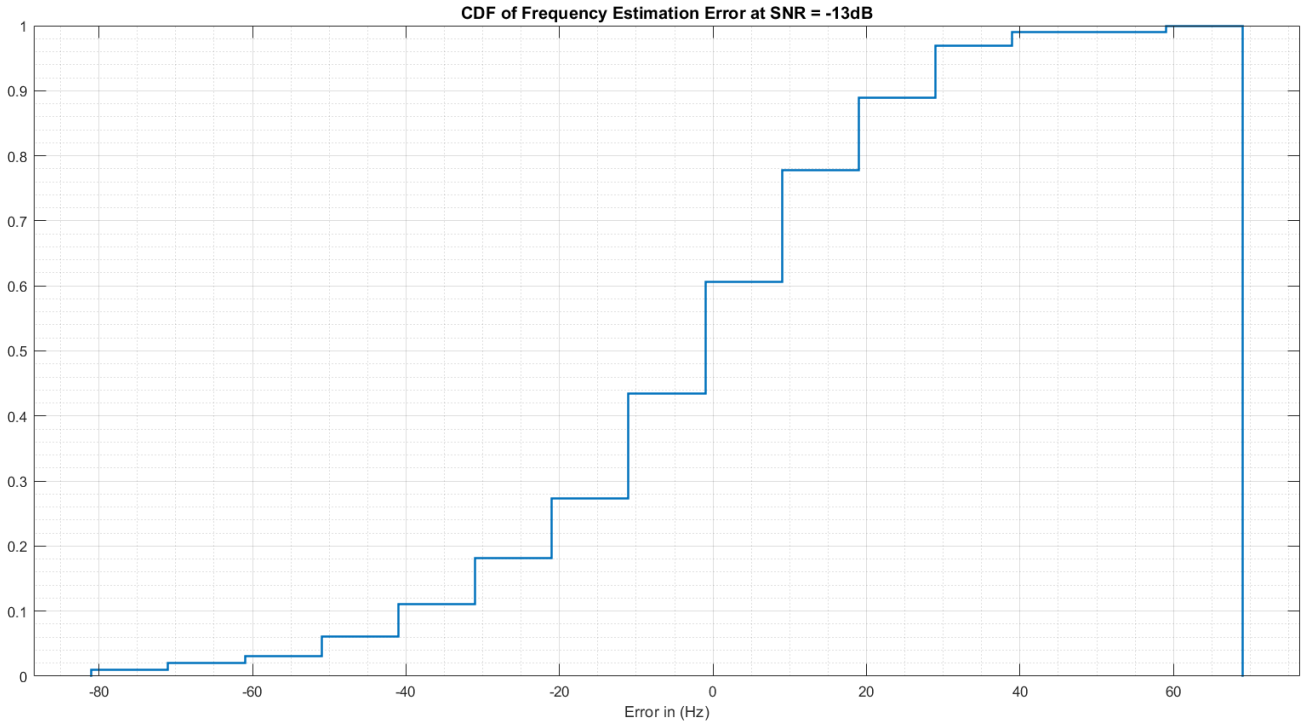


Figure 51 CDF of Frequency Estimation Error in In-Band or Guard-Band Deployment

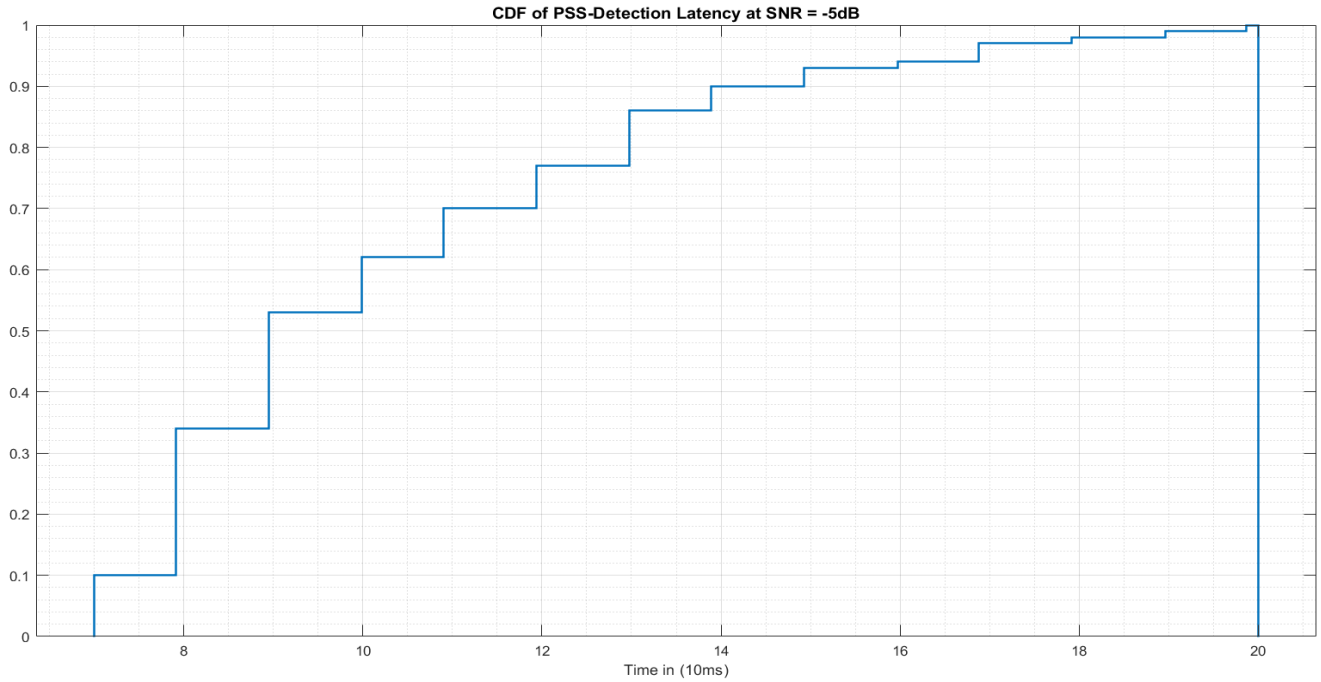


Figure 52 CDF of PSS-Detection Latency in Standalone Deployment

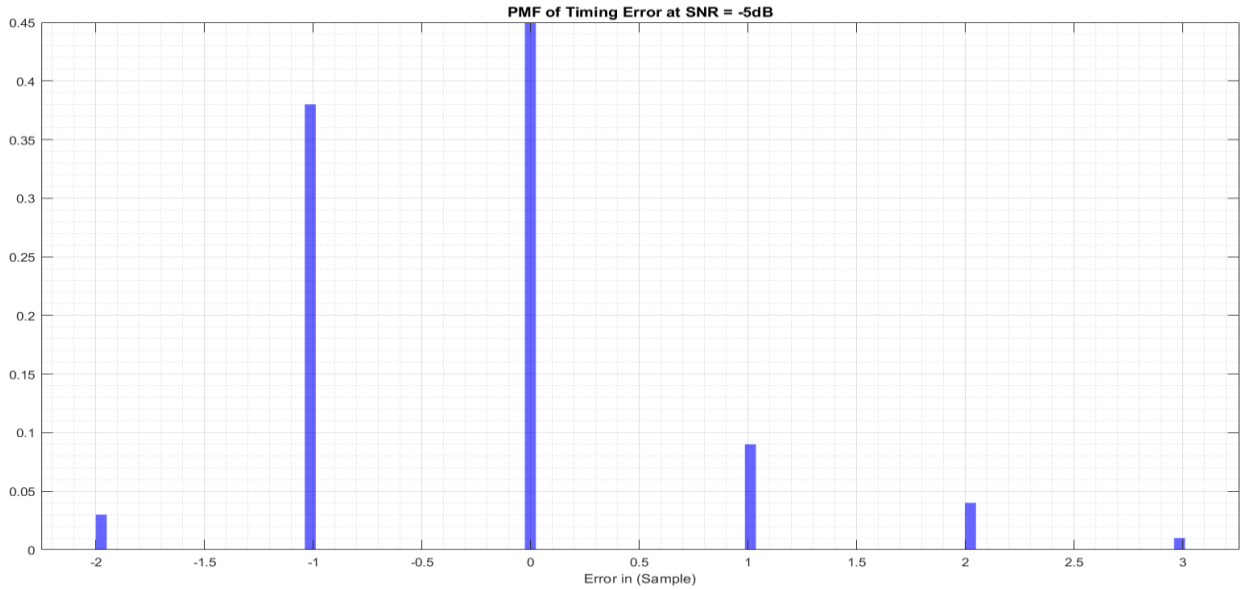


Figure 53 PMF of Synchronization Timing Error in Standalone Deployment

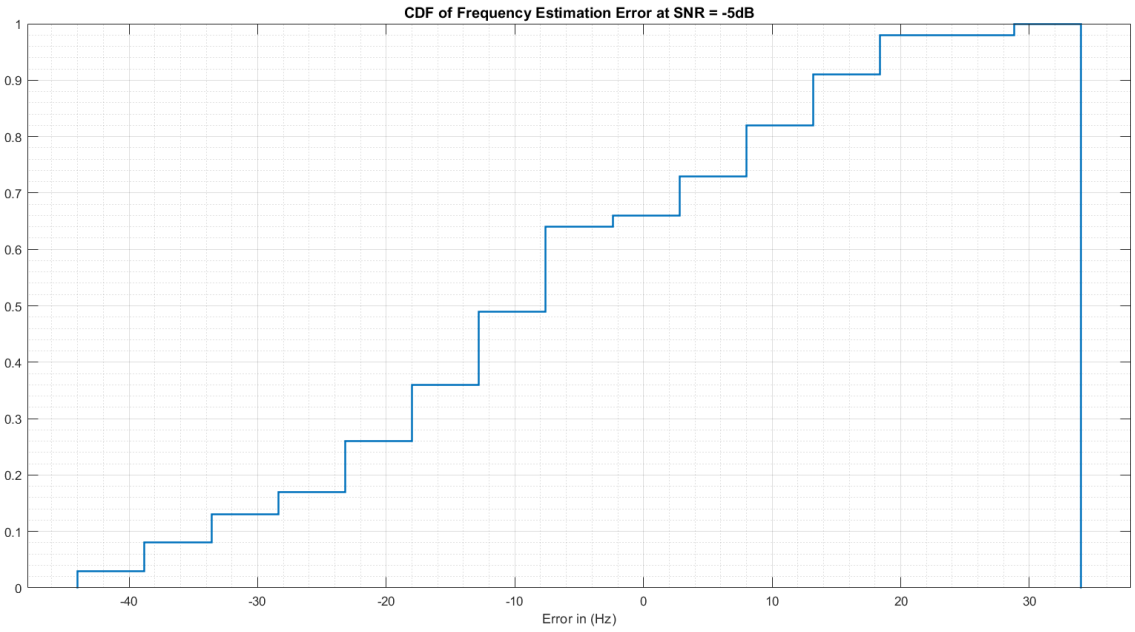


Figure 54 CDF of Frequency Estimation Error in Standalone Deployment

3.1.4.2. Synthesis Results

The Block was synthesized on Design Compiler at technology node 45nm; table 10 summarizes the synthesis results. The synthesis results do not include the shared ram module.

Table 11 Summary of the Synthesis Results for csynch:

Metric	Value
Total Cell-Area	$109445\mu m^2$
Combinational Cell-Area	$89080\mu m^2$
Non-Combinational Cell-Area	$20365\mu m^2$
Critical Path Length	$3.86ns$
Leakage Power	$445\mu w$

3.1.4.3. Screenshots

Illustrated in figures from 55 to 59 screenshots of two runs in MATLAB and ModelSim, the results can vary in estimates between MATLAB and the RTL due to discretization in the RTL. In addition, figure 56 captures a screenshot from the normal operation mode of the block. Figures from 60 to 62 present screenshots from synthesis reports.

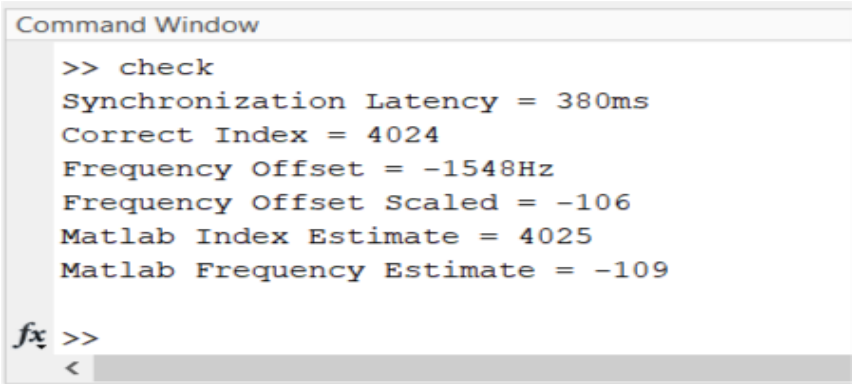


Figure 55 MATLAB Simulation Screenshot for Run 1 for CSYNCH

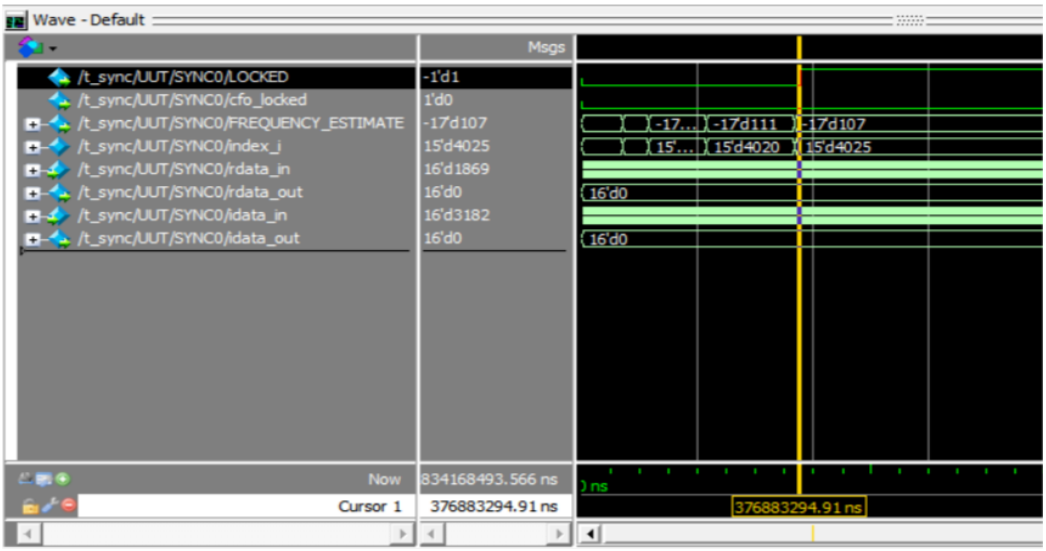


Figure 56 RTL Simulation Screenshot for Run 1 for CSYNCH

```
Command Window
>> check
Synchronization Latency = 520ms
Correct Index = 983
Frequency Offset = -13549Hz
Frequency Offset Scaled = -925
Matlab Index Estimate = 983
Matlab Frequency Estimate = -924

fx >>
```

Figure 57 MATLAB Simulation Screenshot for Run 2 for CSYNCH

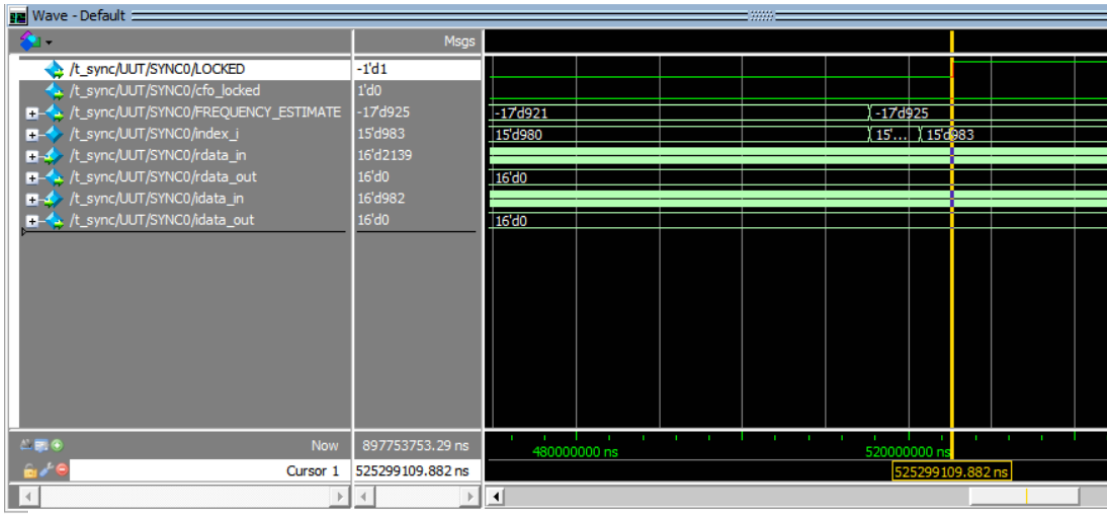


Figure 58 RTL Simulation Screenshot for Run 2 for CSYNCH

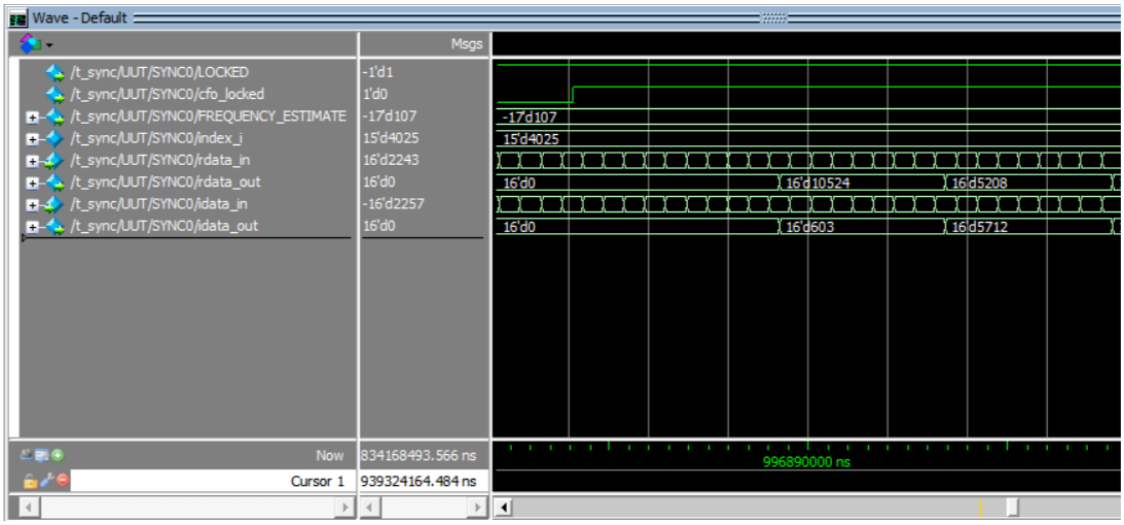


Figure 59 RTL Simulation Screenshot for Normal Operation for CSYNCH

```
Timing Path Group 'clk'
-----------------------------
Levels of Logic:               48.00
Critical Path Length:           3.86
Critical Path Slack:          255.10
Critical Path Clk Period:     520.00
Total Negative Slack:           0.00
No. of Violating Paths:         0.00
Worst Hold Violation:          -0.91
Total Hold Violation:       -3431.48
No. of Hold Violations:      3828.00
-----------------------------------
```

Figure 60 Screenshot from Synthesis QOR Report for CSYNCH

```
Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  41.3250 uW   (94%)
  Net Switching Power  =   2.7772 uW    (6%)
                         ---------
Total Dynamic Power    =  44.1022 uW  (100%)

Cell Leakage Power     = 445.1840 uW

                 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          38.8071        2.1041e-02        6.7803e+04          106.6312  (  21.79%)
sequential         0.0000           0.0000            0.0000            0.0000  (   0.00%)
combinational      2.5184           2.7563         3.7741e+05          382.6853  (  78.21%)
--------------------------------------------------------------------------------------------------
Total             41.3255 uW        2.7774 uW     4.4521e+05 nW       489.3165 uW
1
```

Figure 61 Screenshot from Synthesis Power Report for CSYNCH

```
Number of ports:                          675
Number of nets:                         30028
Number of cells:                        24878
Number of combinational cells:          20950
Number of sequential cells:              3828
Number of macros:                           0
Number of buf/inv:                       5419
Number of references:                     137

Combinational area:       89080.208542
Buf/Inv area:             9182.585949
Noncombinational area:    20364.960657
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:          109445.169199
Total area:                 undefined
1
```

Figure 62 Screenshot from Synthesis Area Report for CSYNCH

3.2. CFO correction:

3.2.1. Block Diagram:

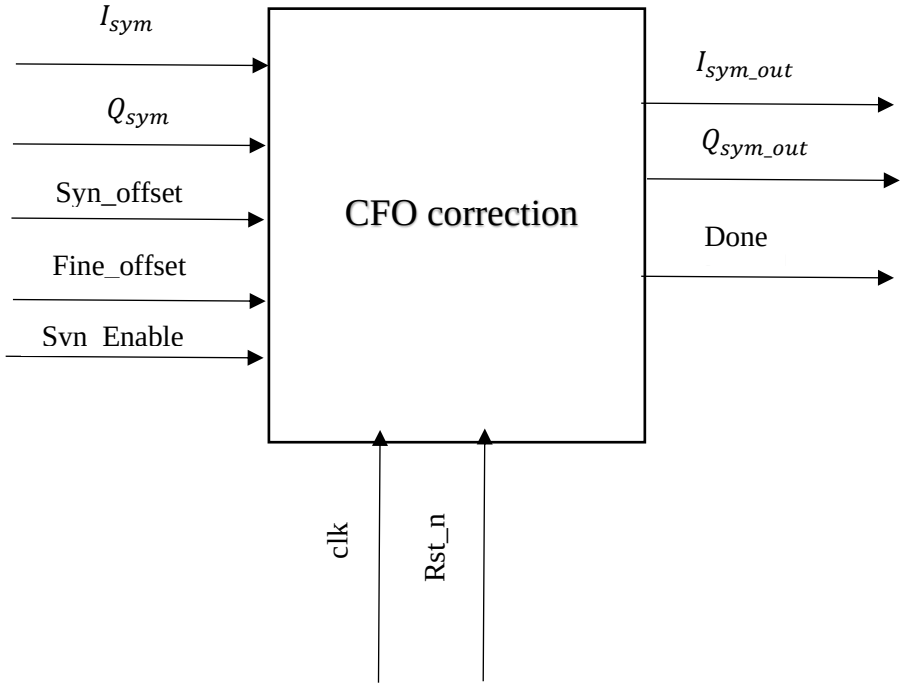


Figure 63 CFO Block Diagram

3.2.2. Interface Table:

Table 12 interface table of CFO correction

Signal Name	Direction	Width	Description
I_{sym}	Input	16	Symbol real part
Q_{sym}	Input	16	Symbol imaginary part
Syn_Offset	Input	22	Synchronization Offset error
Fine_Offset	Input	1	Fine synchronization offset error
Syn_Enable	Input	1	CFO enable
clk	Input	1	CFO clock
Rst_n	Input	1	CFO reset
I_{sym_out}	Output	16	corrected symbol real part
Q_{sym_out}	Output	16	corrected symbol imaginary part
Done	Output	1	Signal indicate that block is done

3.2.3. Function of the design:

- Rotate the symbol to correct the CFO using CORDIC algorithms using the offset that come out from phase accumulator.

3.2.4. Design specification:

- The clock used for the block has period equal 260 ns.
- Latency of the block is 16 clock cycle.

3.2.5. Detailed block diagram:

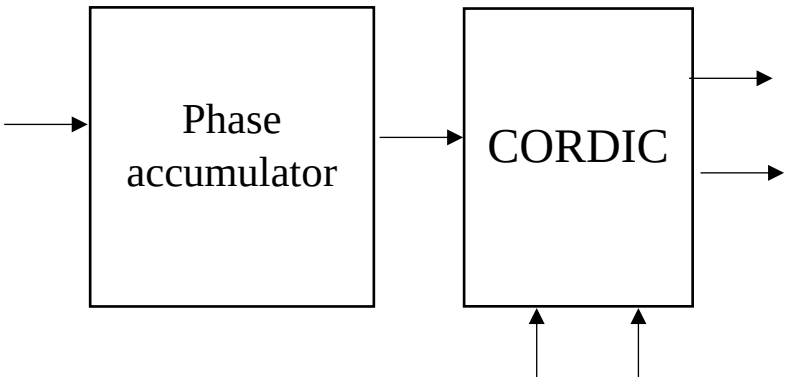


Figure 64 CFO Detailed Block Diagram

3.2.5.1. Phase accumulator

Phase accumulator used to accumulate the offset as initially the offset that get out from phase accumulator equal 0 then it accumulates every clock with the value of the offset that come from synchronization block until the offset exceeds 2pi the new value will be equal to offset-2pi and so on.

3.2.5.2. CORDIC

CORDIC is used to rotate the symbol to correct the offset occurred. To achieve a high accuracy, we should make n iterations as we mentioned before as iterations increase the accuracy increase but we can achieve an acceptable accuracy by 15 iterations.

3.2.5.2.1.1. CORDIC design

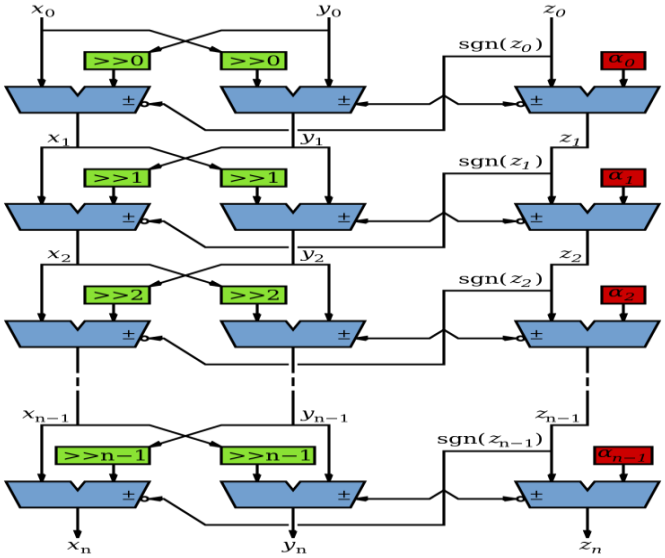


Figure 65 : CORDIC block diagram

But this design has many disadvantages:

- Consumes high power which contradict with the project that should be low power.
- Large area as there is n shifters and adders.

We reached to a design that has lower area and consumes lower power than the first design.

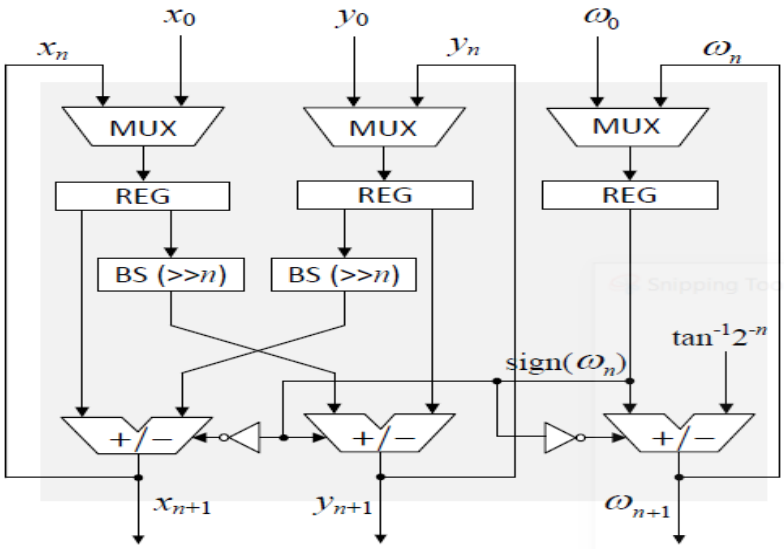


Figure 66 recursive pipelined CORDIC design

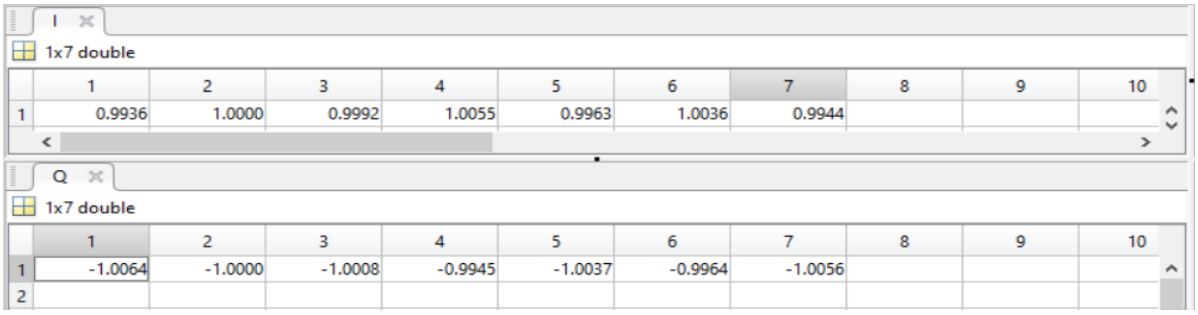
- This design has lower number of shifters and adders n times than first design.
- But it has lower throughput.

3.2.6. Design Interface:

- The interface of the block with the synchronization block is the 16-bit real and imaginary part of the symbol and 22-bit of the offset and 1-bit enable signal to make the block start of operate.
- The interface of the block with the fine synchronization block is 22-bit of the fine offset.
- The interface of the block with the FFT block is the 16-bit real and imaginary part of the corrected symbol and 1-bit done signal.

3.2.7. Simulation Results:

3.2.7.1. MATLAB Results:


I
1x7 double

	1	2	3	4	5	6	7	8	9	10
1	0.9936	1.0000	0.9992	1.0055	0.9963	1.0036	0.9944			

Q
1x7 double

	1	2	3	4	5	6	7	8	9	10
1	-1.0064	-1.0000	-1.0008	-0.9945	-1.0037	-0.9964	-1.0056			
2										

Figure 67 MATLAB result for different input offset CFO

3.2.7.2. RTL Results:

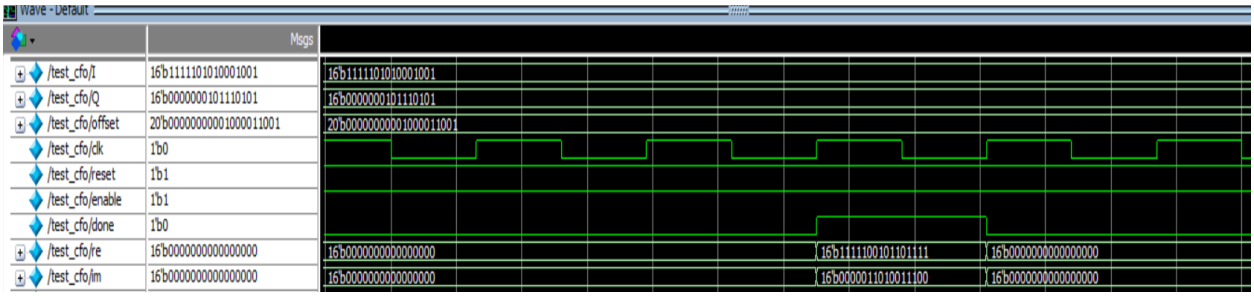


Figure 68 RTL simulation results of CFO

3.2.7.3. Comparison between RTL and MATLAB

By entering the same input in the RTL and MATLAB we get the following result

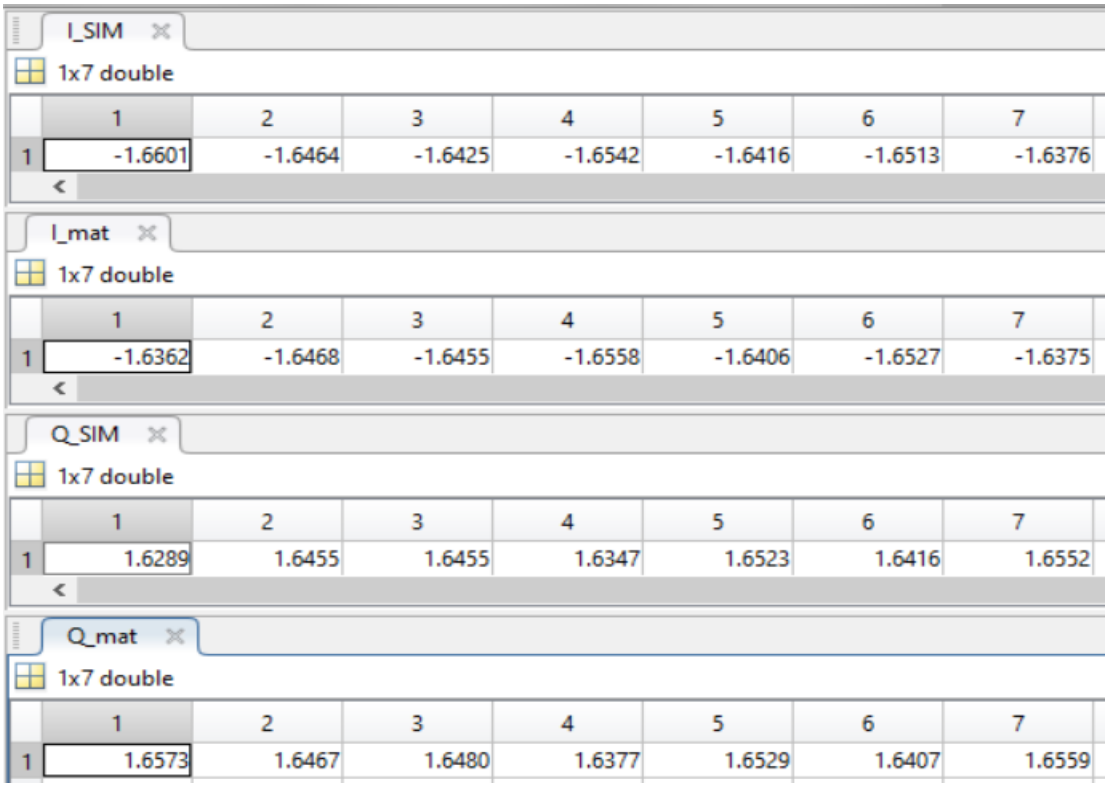


Figure 69: comparing results of RTL and MATLAB in CFO

And by calculating the average error in the real and imaginary

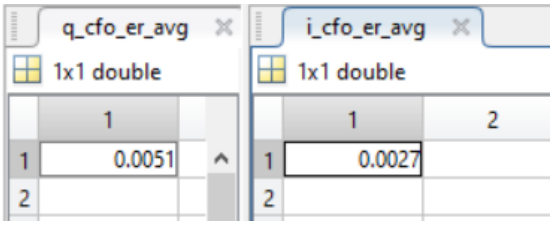


Figure 70: average error between RTL and MATLAB in CFO

3.2.7.4. Synthesis Reports:

```
Number of ports:                          150
Number of nets:                          1341
Number of cells:                          939
Number of combinational cells:            798
Number of sequential cells:               129
Number of macros:                           0
Number of buf/inv:                        149
Number of references:                      42

Combinational area:       1822.898004
Buf/Inv area:              121.562000
Noncombinational area:     647.976006
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:          2470.874010
Total area:                 undefined
1
```

Figure 71: Area report of CFO

```
Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =   3.1238 uW   (73%)
  Net Switching Power  =   1.1418 uW   (27%)
                         ---------
Total Dynamic Power    =   4.2657 uW  (100%)

Cell Leakage Power     =  11.3075 uW


                 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           2.2702            0.1230        2.2595e+03            4.6526
(  29.88%)
sequential         0.0000            0.0000            0.0000            0.0000
(   0.00%)
combinational      0.8536            1.0189        9.0481e+03           10.9205
(  70.12%)
--------------------------------------------------------------------------------
------------------
Total              3.1238 uW         1.1418 uW     1.1308e+04 nW        15.5732
uW
1
```

Figure 72 : Power report of CFO

```
  clock clk (rise edge)                                  260.42     260.42
  clock network delay (ideal)                              0.00     260.42
  clock uncertainty                                       -0.50     259.92
  offset_final_reg[12]/CK (DFF_X1)                         0.00     259.92 r
  library setup time                                      -0.04     259.88
  data required time                                                259.88
  ---------------------------------------------------------------------------
  data required time                                                259.88
  data arrival time                                                -133.33
  ---------------------------------------------------------------------------
  slack (MET)                                                       126.55


1
```

Figure 73: Timing report of CFO

3.3. FFT

3.3.1. Block Diagram:

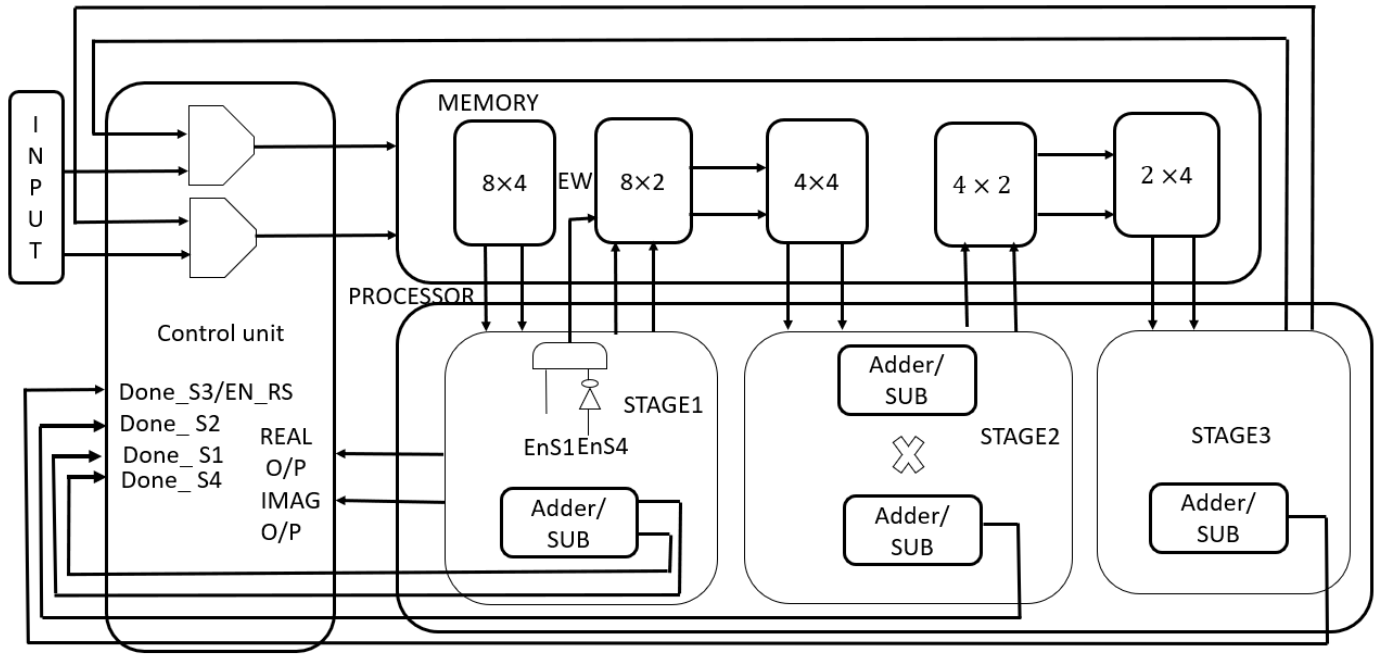


Figure 74 16-point FFT radix2^2 SDF block diagram

3.3.2. Interface Table:

Table 13 interface table of FFT

Signal Name	Direction	Width	Description
clk_260	Input	1-bit	Input clock
Reset	Input	1bit	Asynchronous reset

Enable_CFO	Input	1-bit	Enable from the FIFO before the FFT to enable the FFT
realINPUT_CU	Input	16-bit	The real input symbols
imagINPUT_CU	Input	16-bit	The imaginary input symbols
nf_SYNC_CU	Input	10-bit	NF is passing through the FFT to reach fine in the proper time along with the data
ns_SYNC_CU	Input	5-bit	NS is passing through the FFT to reach fine in the proper time along with the data
nf_FFT_CU	Output	10-bit	NF output
ns_FFT_CU	Output	5-bit	NS output
done_FFT_CU	Output	1-bit	FFT is done to enable the resource mapper
enable_RS_CU	Output	1-bit	Enable signal indicates that the FFT is done and the resource de-mapper can start saving the FFT output
S1_outputR_CU	Output	16-bit	The real output symbols
S1_outputI_CU	Output	16-bit	The real output symbols

3.3.3. Function of the design:

i. Before optimization

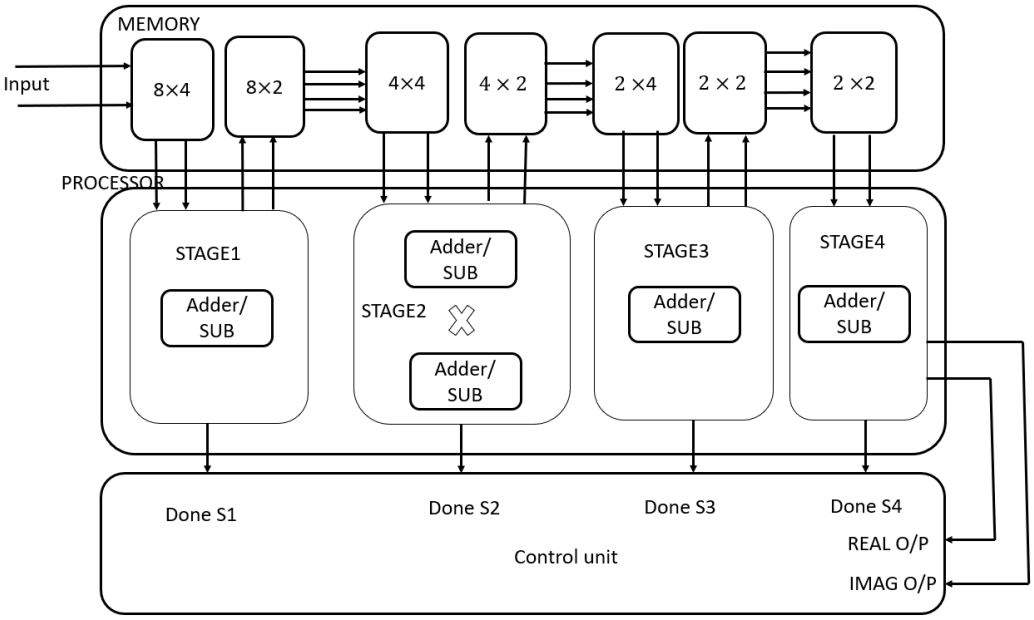


Figure 75 16-point FFT Block diagram before optimized

Data flow through stages:

Firstly, the 16 OFDM symbols are stored in 8×4 memory and to improve system latency the first stage will start to operate as soon as the nineth symbols stored in the memory. The output of this stage is stored in

8× 2 memory when the eight symbols are ready, they will transfer parallel to 4×4 memory and the second stage arithmetic operation starts to proceed then the output of this operation is stored in memory 4×2 and then transferred parallelly to 2×4 memory and enable signal to stage 3 is raised. Similarly, stage 3 starts doing the arithmetic operation and store the output in memory 2×2 which is then be copied parallelly to another 2×2 memory and an enable to stage 4 is raised to start operating and the output are stored in the resource element de-mapper.

In this design, the 3D memory is used in order to optimize the address generating circuit, that's was accomplished by storing the values which the arithmetic operation will be executed in the same row so by using only one counter, we get the data from the memory.

The size of the memories depends on the algorithm RADIX 2^2 for instance in second stage, the first symbol is added to the fifth one and the second is added to sixth one and the third data is added to the seventh one and the fourth is added on the eighth one. Then the required memory is 8-words memory for real and other 8-words memory for imaginary.

The bit reverse operation which is important as the FFT output is out-of-order. This operation is done while storing in the resource element de-mapper.

ii. After optimization

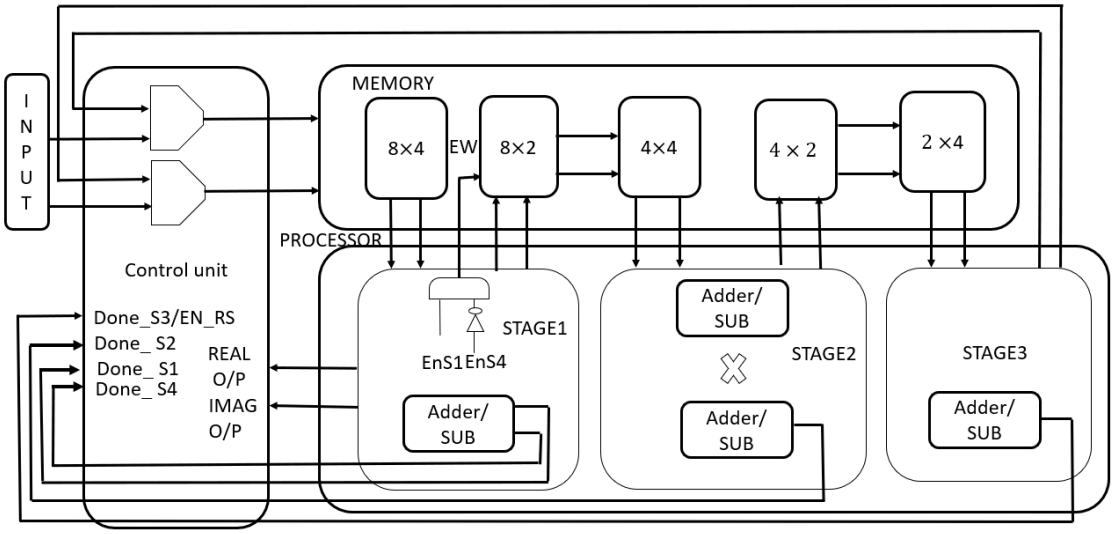


Figure 76 Optimized 16-point FFT block diagram

The algorithm has further been optimized by sharing the resource between stage one and stage four; the first stage is done before the stage four start. As noticed from Figure 75 and Figure 76, both 2×2 memory and stage 4 are no longer in the block diagram that has been merged and a MUX was used, that multiplex the input of FFT and the third stage output as the FFT input is entered serially to the block so less number of MUXs were used compared to parallel input way. An enable signal is used to store only the first stage output and avoid storing the fourth stage output as it is the FFT output. The register file is used instead of the shift register in order to reduce the switching. The shift reg is used to optimize the power by reducing the switching activity.

3.3.4. Detailed implementation:

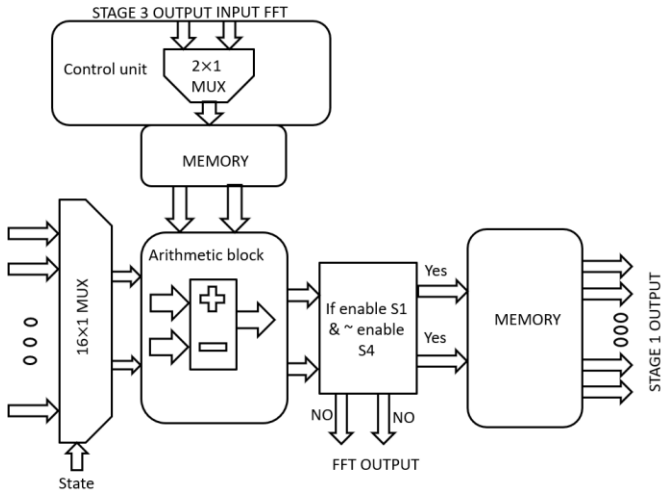


Figure 77 Stage 1&4 after optimization block diagram of FFT

The 16 mux is used to decide which state, of the sixteen states, is next. Depending on the stage the control signal is sent to the arithmetic block which is designed to be a combinational block to do the operation once the inputs and control signal come within a clock cycle.

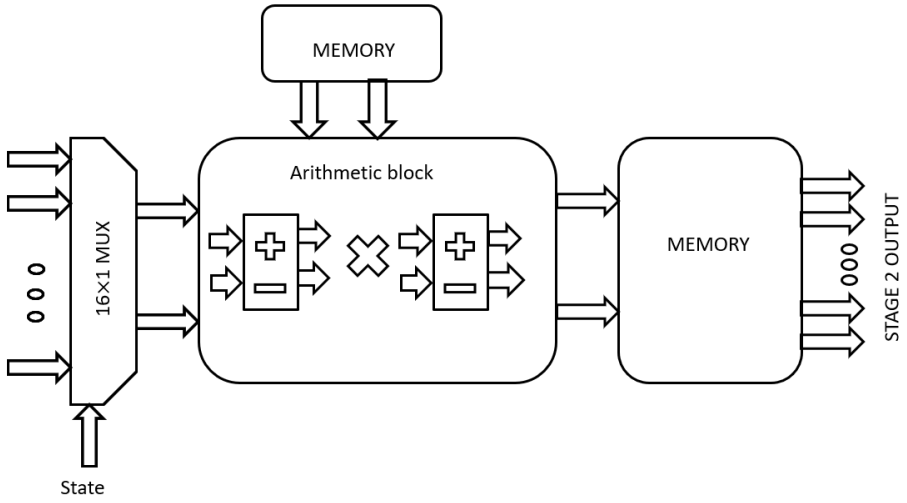


Figure 78 Stage 2 block diagram of FFT

When the enable signal “S2” is high that means the input is ready. Then the state and the control signal will be used to determine the next arithmetic operation and the value of the twiddle factors. The arithmetic block consists of two adder/subtractor and one multiplier. There is a mux between the output of adder/subtractor and the input of the arithmetic block and there is another mux between the multiplier and the input of the arithmetic block.

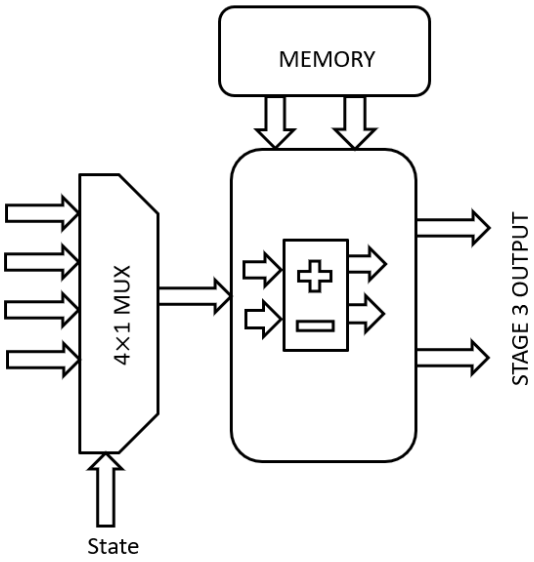


Figure 79 Stage 3 block diagram of FFT

Stage 3 starts when the fourth output of stage 2 is ready so the enable signal “S3” is high. Bear in mind, the mux used here is 4×1 as there is only four states and they are repeated one after the other.

3.3.5. Design Interface:

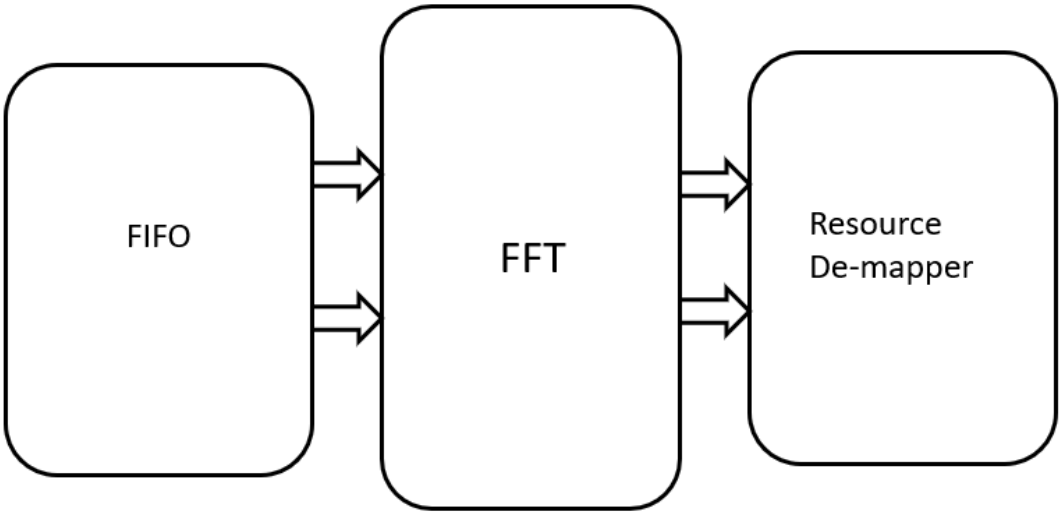


Figure 80 interfacing blocks of FFT

The CFO gives output each sixteen-clock cycle which is not suitable for the implemented algorithm of FFT, thus a FIFO is added to store 16 symbols then gives enable to FFT to start it operation the output of this buffer is serial output. The output of the FFT is saved in Resource mapper, and the output is out serially.

3.3.6. Design Specification

- Clock used is 260ns
- Block latency is 38 clock cycle

3.3.7. Simulation Results:

3.3.7.1. MATLAB Results:

```
Columns 1 through 7

  1.3600 + 1.3600i   0.3024 + 0.3024i  -0.0836 - 0.0836i  -0.4036 - 0.4036i   0.2400 + 0.2400i   0.0387 + 0.0387i  -0.0660 - 0.0660i

Columns 8 through 14

 -0.3856 - 0.3856i  -0.0800 - 0.0800i   0.1983 + 0.1983i  -0.0767 - 0.0767i  -0.3964 - 0.3964i  -0.4000 - 0.4000i   0.4206 + 0.4206i

Columns 15 through 16

 -0.0937 - 0.0937i  -0.4143 - 0.4143i
```

Figure 81 16-point FFT MATLAB output of FFT

```
Columns 1 through 7

 -0.0767 - 0.0767i  -0.3964 - 0.3964i  -0.4000 - 0.4000i   0.4206 + 0.4206i  -0.0937 - 0.0937i  -0.4143 - 0.4143i   1.3600 + 1.3600i

Columns 8 through 12

  0.3024 + 0.3024i  -0.0836 - 0.0836i  -0.4036 - 0.4036i   0.2400 + 0.2400i   0.0387 + 0.0387i
```

Figure 82 12 FFT output stored in resource de-mapper of FFT

3.3.7.2. RTL Results:

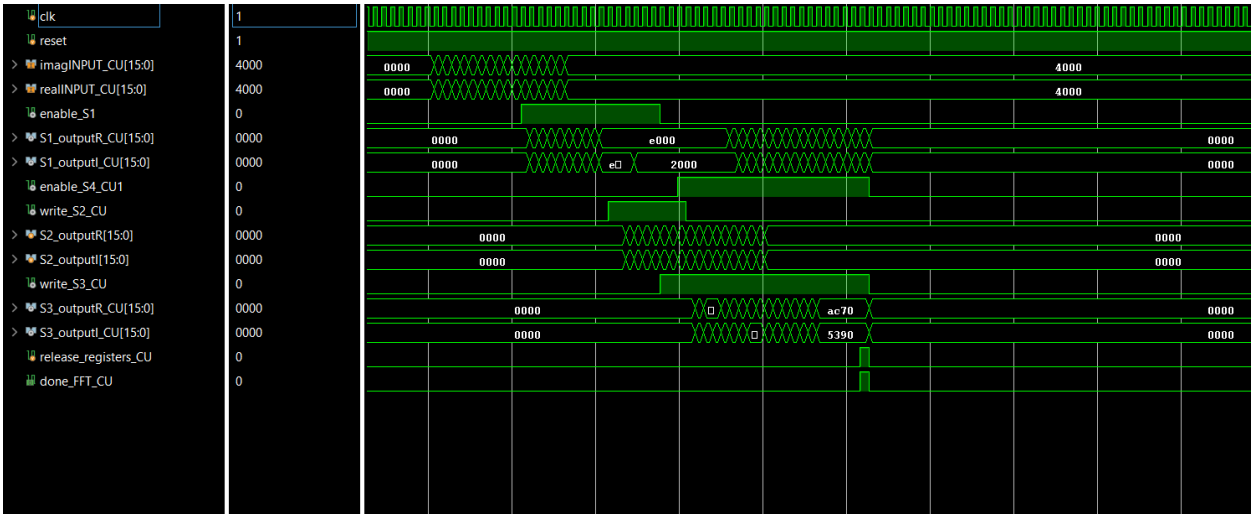


Figure 83 16-point FFT RTL simulation of FFT

FIFO simulation

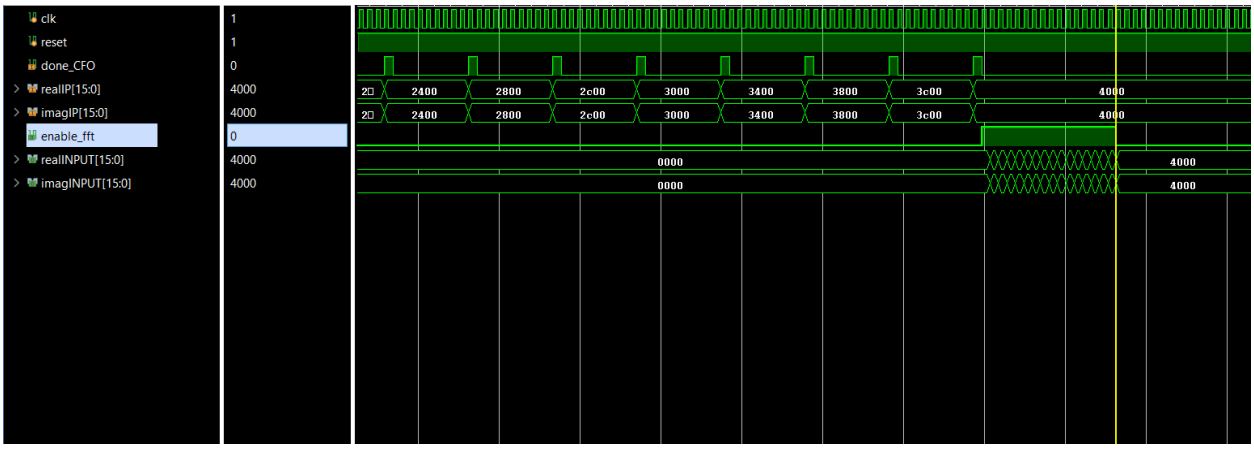


Figure 84 FIFO RTL simulation of FFT

3.3.7.3. Synthesis Reports:

i. Screenshots for synthesis results

```
CFO2FFT_memory         5K_hvratio_1_1    NangateOpenCellLibrary_ss0p95vn40c


Global Operating Voltage = 0.95
Power-specific unit information :
   Voltage Units = 1V
   Capacitance Units = 1.000000ff
   Time Units = 1ns
   Dynamic Power Units = 1uW    (derived from V,C,T units)
   Leakage Power Units = 1nW


  Cell Internal Power  =   3.1194 uW   (93%)
  Net Switching Power  = 242.3862 nW    (7%)
                         ---------
Total Dynamic Power    =   3.3618 uW  (100%)

Cell Leakage Power     =  20.6408 uW


                 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           3.0167        1.7397e-02        9.5064e+03           12.5405  (  52.25%)
sequential         0.0000           0.0000            0.0000             0.0000  (   0.00%)
combinational      0.1027           0.2250        1.1134e+04           11.4621  (  47.75%)
--------------------------------------------------------------------------------------------------
Total              3.1194 uW        0.2424 uW     2.0641e+04 nW        24.0026 uW
1
```

Figure 85 Power Report of memory before FFT

```
Design        Wire Load Model            Library
------------------------------------------------
FFT_controlunit        5K_hvratio_1_1    NangateOpenCellLibrary_ss0p95vn40c


Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  21.1035 uW   (95%)
  Net Switching Power  =   1.1915 uW    (5%)
                         ---------
Total Dynamic Power    =  22.2950 uW  (100%)

Cell Leakage Power     =  99.3396 uW


                 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          20.6093            0.1088        3.7598e+04           58.3159  (  47.94%)
sequential         0.0000            0.0000            0.0000            0.0000  (   0.00%)
combinational      0.4942            1.0827        6.1742e+04           63.3187  (  52.06%)
--------------------------------------------------------------------------------------------------
Total             21.1035 uW         1.1915 uW     9.9340e+04 nW       121.6345 uW
1
```

Figure 86 Power Report of FFT

```
Number of ports:                         98
Number of nets:                        2825
Number of cells:                       2207
Number of combinational cells:         1638
Number of sequential cells:             569
Number of macros:                         0
Number of buf/inv:                      549
Number of references:                    18

Combinational area:       2092.888009
Buf/Inv area:              405.383991
Noncombinational area:    2577.805909
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:          4670.693918
Total area:                 undefined
1
```

Figure 87 Area Report of memory before FFT

```
Number of ports:                          100
Number of nets:                           624
Number of cells:                           77
Number of combinational cells:             74
Number of sequential cells:                 0
Number of macros:                           0
Number of buf/inv:                         40
Number of references:                       8

Combinational area:       11717.566066
Buf/Inv area:             1639.357978
Noncombinational area:    11173.330360
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:          22890.896427
Total area:                 undefined
1
```

Figure 88 Area Report of FFT

```
clock clk_260 (rise edge)                                 520.00     520.00
clock network delay (ideal)                                 0.00     520.00
clock uncertainty                                          -0.35     519.65
memory_beforeS1_imag_reg[2][6]/CK (DFF_X1)                  0.00     519.65 r
library setup time                                         -0.04     519.61
data required time                                                   519.61
---------------------------------------------------------------------------
data required time                                                   519.61
data arrival time                                                    -1.57
---------------------------------------------------------------------------
slack (MET)                                                          518.05
```

Figure 89 Timing Report of memory before FFT

```
clock clk_260 (rise edge)                                 520.00     520.00
clock network delay (ideal)                                 0.00     520.00
clock uncertainty                                          -0.35     519.65
S2/memory_afterS2_reg[1][0][14]/CK (DFFR_X1)                0.00     519.65 r
library setup time                                         -0.04     519.61
data required time                                                   519.61
---------------------------------------------------------------------------
data required time                                                   519.61
data arrival time                                                    -3.37
---------------------------------------------------------------------------
slack (MET)                                                          516.24
```

Figure 90 Timing Report of FFT

ii. Comparison for synthesis results with [25]

Table 14 FFT comparison with [25]

FFT comparison			
Areas in [25] technology node 0.18 μm	Design area technology node 45nm	Power in [25] technology node 0.18μm	Design power technology node 45nm
0.5314mm^2	22890.896427μm^2	22.4mW	121.6345 μW

3.4. RESOURCE DE-MAPPER

3.4.1. Block Diagram:

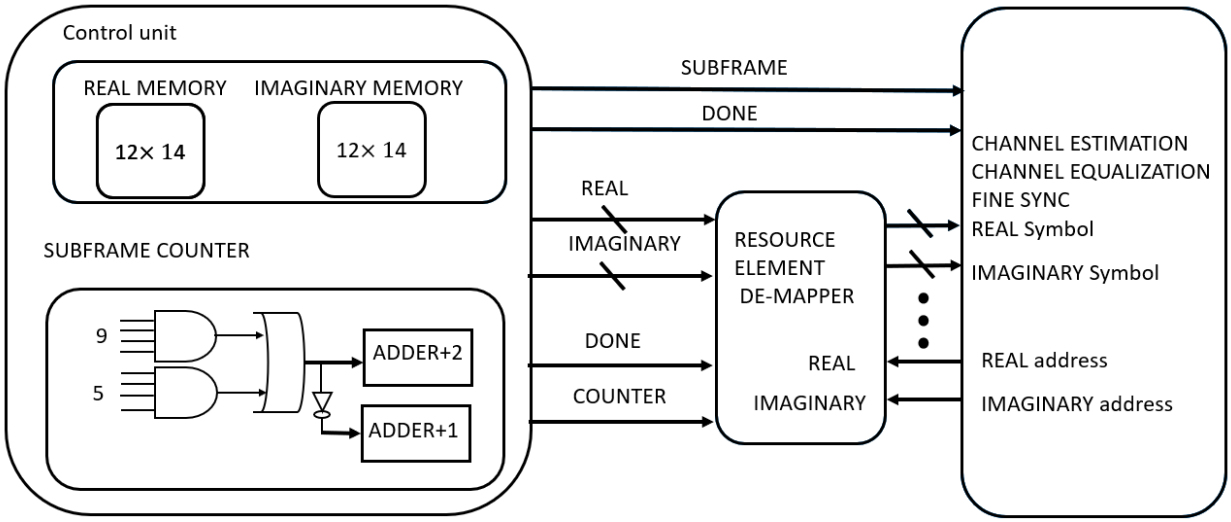


Figure 91 resource element de-mapper block diagram

3.4.2. Interface Table:

Storage Element memory

Table 15 Storage Element Interface Table

Signal Name	Direction	Width	Description
clk_260	INPUT	1-bit	Input clock to block
Reset	INPUT	1-bit	Reset signal
done_FFT	INPUT	1-bit	The control signal that indicates that FFT output is ready and resource de-mapper can start operation
RS_INPUT_REAL	INPUT	16-bit	Input real data to resource de-mappper
RS_INPUT_IMAG	INPUT	16-bit	Input imaginary data to resource de-mappper
Done_RS	OUTPUT	1-bit	The resource de-mapper is full

RS_subframe	OUTPUT	1-bit	Subframe number
done_ResourceDemapper	OUTPUT	1-bit	Resource de-mapper has the subframe ready
Read_REM	OUTPUT	1-bit	Enable to resource de-mapper
Read_REM_counter	OUTPUT	4-bit	Counter from 0 to 11 to store in the resource de-mapper
input_resourceR0	OUTPUT	16-bit	Real output parallel of index 0
input_resourceR1	OUTPUT	16-bit	Real output parallel of index 1
input_resourceR2	OUTPUT	16-bit	Real output parallel of index 2
input_resourceR3	OUTPUT	16-bit	Real output parallel of index 3
input_resourceR4	OUTPUT	16-bit	Real output parallel of index 4
input_resourceR5	OUTPUT	16-bit	Real output parallel of index 5
input_resourceR6	OUTPUT	16-bit	Real output parallel of index 6
input_resourceR7	OUTPUT	16-bit	Real output parallel of index 7
input_resourceR8	OUTPUT	16-bit	Real output parallel of index 8
input_resourceR9	OUTPUT	16-bit	Real output parallel of index 9
input_resourceR10	OUTPUT	16-bit	Real output parallel of index 10
input_resourceR11	OUTPUT	16-bit	Real output parallel of index 11
input_resourceI0	OUTPUT	16-bit	Imaginary output parallel of index 0
input_resourceI1	OUTPUT	16-bit	Imaginary output parallel of index 1
input_resourceI2	OUTPUT	16-bit	Imaginary output parallel of index 2
input_resourceI3	OUTPUT	16-bit	Imaginary output parallel of index 3
input_resourceI4	OUTPUT	16-bit	Imaginary output parallel of index 4
input_resourceI5	OUTPUT	16-bit	Imaginary output parallel of index 5
input_resourceI6	OUTPUT	16-bit	Imaginary output parallel of index 6
input_resourceI7	OUTPUT	16-bit	Imaginary output parallel of index 7
input_resourceI8	OUTPUT	16-bit	Imaginary output parallel of index 8
input_resourceI9	OUTPUT	16-bit	Imaginary output parallel of index 9
input_resourceI10	OUTPUT	16-bit	Imaginary output parallel of index 10
input_resourceI11	OUTPUT	16-bit	Imaginary output parallel of index 11

Resource De-mapper

Table 16 Resource De-mapper interface table

Signal Name	Direction	Width	Description
clk_260	INPUT	1-bit	Input clock to block
Reset	INPUT	1-bit	Reset signal
done_FFT	INPUT	1-bit	The control signal that indicates that FFT output is ready and resource de-mapper can start operation
Read_REM	INPUT	1-bit	Enable to resource de-mapper
Read_REM_counter	INPUT	4-bit	Counter from 0 to 11 to store in the resource de-mapper
input_resourceR0	INPUT	16-bit	Real output parallel of index 0
input_resourceR1	INPUT	16-bit	Real output parallel of index 1
input_resourceR2	INPUT	16-bit	Real output parallel of index 2
input_resourceR3	INPUT	16-bit	Real output parallel of index 3
input_resourceR4	INPUT	16-bit	Real output parallel of index 4
input_resourceR5	INPUT	16-bit	Real output parallel of index 5

input_resourceR6	INPUT	16-bit	Real output parallel of index 6
input_resourceR7	INPUT	16-bit	Real output parallel of index 7
input_resourceR8	INPUT	16-bit	Real output parallel of index 8
input_resourceR9	INPUT	16-bit	Real output parallel of index 9
input_resourceR10	INPUT	16-bit	Real output parallel of index 10
input_resourceR11	INPUT	16-bit	Real output parallel of index 11
input_resourceI0	INPUT	16-bit	Imaginary output parallel of index 0
input_resourceI1	INPUT	16-bit	Imaginary output parallel of index 1
input_resourceI2	INPUT	16-bit	Imaginary output parallel of index 2
input_resourceI3	INPUT	16-bit	Imaginary output parallel of index 3
input_resourceI4	INPUT	16-bit	Imaginary output parallel of index 4
input_resourceI5	INPUT	16-bit	Imaginary output parallel of index 5
input_resourceI6	INPUT	16-bit	Imaginary output parallel of index 6
input_resourceI7	INPUT	16-bit	Imaginary output parallel of index 7
input_resourceI8	INPUT	16-bit	Imaginary output parallel of index 8
input_resourceI9	INPUT	16-bit	Imaginary output parallel of index 9
input_resourceI10	INPUT	16-bit	Imaginary output parallel of index 10
input_resourceI11	INPUT	16-bit	Imaginary output parallel of index 11
channel_estimation_address_row0	INPUT	4-bit	Channel estimation address for first row
channel_estimation_address_column0	INPUT	4-bit	Channel estimation address for first column
channel_estimation_address_row1	INPUT	4-bit	Channel estimation address for second row
channel_estimation_address_column1	INPUT	4-bit	Channel estimation address for second column
channel_equalization_address_column1	INPUT	4-bit	Channel equalization address column
fine_address_row	INPUT	4-bit	Fine address row
fine_address_column	INPUT	4-bit	Fine address column
channel_estimation_Real_0	OUTPUT	16-bit	Channel estimation real data of index 0
channel_estimation_imag_0	OUTPUT	16-bit	Channel estimation imaginary data of index 0
channel_estimation_real_1	OUTPUT	16-bit	Channel estimation real data of index 1
channel_estimation_imag_1	OUTPUT	16-bit	Channel estimation imaginary data of index 1
channel_equalization_real_0	OUTPUT	16-bit	Channel equalization real data of index 0
channel_equalization_imag_0	OUTPUT	16-bit	Channel equalization imaginary data of index 0
channel_equalization_real_1	OUTPUT	16-bit	Channel equalization real data of index 1
channel_equalization_imag_1	OUTPUT	16-bit	Channel equalization imaginary data of index 1

channel_equalization_real_2	OUTPUT	16-bit	Channel equalization real data of index 2
channel_equalization_imag_2	OUTPUT	16-bit	Channel equalization imaginary data of index 2
channel_equalization_real_3	OUTPUT	16-bit	Channel equalization real data of index 3
channel_equalization_imag_3	OUTPUT	16-bit	Channel equalization imaginary data of index 3
channel_equalization_real_4	OUTPUT	16-bit	Channel equalization real data of index 4
channel_equalization_imag_4	OUTPUT	16-bit	Channel equalization imaginary data of index 4
channel_equalization_real_5	OUTPUT	16-bit	Channel equalization real data of index 5
channel_equalization_imag_5	OUTPUT	16-bit	Channel equalization imaginary data of index 5
channel_equalization_real_6	OUTPUT	16-bit	Channel equalization real data of index 6
channel_equalization_imag_6	OUTPUT	16-bit	Channel equalization imaginary data of index 6
channel_equalization_real_7	OUTPUT	16-bit	Channel equalization real data of index 7
channel_equalization_imag_7	OUTPUT	16-bit	Channel equalization imaginary al data of index 7
channel_equalization_real_8	OUTPUT	16-bit	Channel equalization real data of index 8
channel_equalization_imag_8	OUTPUT	16-bit	Channel equalization imaginary data of index 8
channel_equalization_real_9	OUTPUT	16-bit	Channel equalization real data of index 9
channel_equalization_imag_9	OUTPUT	16-bit	Channel equalization imaginary data of index 9
channel_equalization_real_10	OUTPUT	16-bit	Channel equalization real data of index 10
channel_equalization_imag_10	OUTPUT	16-bit	Channel equalization imaginary data of index 10
channel_equalization_real_11	OUTPUT	16-bit	Channel equalization real data of index 11
channel_equalization_imag_11	OUTPUT	16-bit	Channel equalization imaginary data of index 11
fine_real	OUTPUT	16-bit	Fine real output
fine_imag	OUTPUT	16-bit	Fine imaginary output

3.4.3. Function of the design:

The block is divided into two modules the first one is a storage element memory consists of a memory 16×14 that stores the output of the FFT and when a whole subframe is ready an enable signal is raised to let the resource element de-mapper get the symbols column by column so it takes 14 clock cycle to fill the resource de-mapper. Thus, the time in which the resource de-mapper is kept unchanged for approximately 1ms as stated in the standard.[5]

The main reason for having the storage element memory is that it takes 128 clocks cycle to get a new OFDM symbol, so the data would change in less than 1ms and it is important for blocks that follow the resource de-mapper performance as they need the data to stay for a period of time till they finish their operation. For instance, the channel equalization is working after the channel estimation is done so if the data in the resource de-mapper is changed the channel estimation cannot work correctly.

The channel estimation, channel equalization, fine can access the memory using the row and column address and the process of accessing the memory with the address and get the data works as a combinational process.

	subframe number									
even numbered frame	0	1	2	3	4	5	6	7	8	9
	NPBCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPSS	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NSSS
	subframe number									
odd numbered frame	0	1	2	3	4	5	6	7	8	9
	NPBCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPSS	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH

Figure 92 Time multiplexing between NB-IoT downlink physical channels and signals [3]

The subframe 5 contains the NPSS as shown in fig 5 used by the coarse synchronization so it is not passing through the chain and subframe 9 which contains the NSSS is also not passing through the chain, as well. Therefore, the subframe number is 0, 1, 2, 3, 4, 6, 7, 8 instead of from 0 to 9.

In the storage element memory, the bit-reversing operation for the FFT output takes place, so the symbols stored in this memory are in-order. Then, when transferring the data from storage element memory to resource element de-mapper the symbols are resorted as the last six symbols stored in first six symbols and the first six symbols are stored in last six places in memory as shown in table 3.

Table 17 Bit-reversing operation

FFT output out-of order	**FFT output**	**Index in which data stored in in the Resource De-mapper**
0	0	6
1	8	
2	4	10
3	12	2
4	2	8
5	10	0
6	6	
7	14	4
8	1	7
9	9	
10	5	11
11	13	3
12	3	9
13	11	1
14	7	
15	15	5

3.4.4. Design Interface:

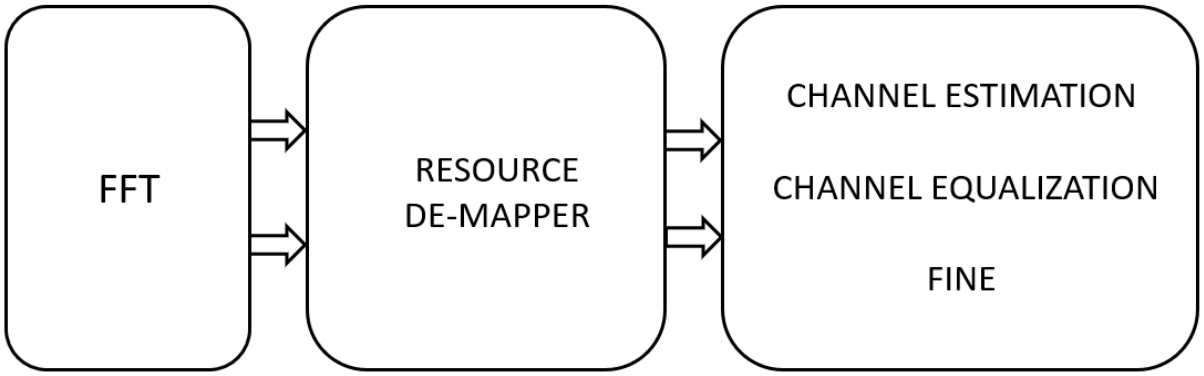


Figure 93 interface block diagram

The resource de-mapper is taking the output of the FFT then the output is stored in its memory to be then used by the channel estimation, channel equalization and fine using the row and column address.

3.4.5. Design Specification

- Clock used is 260ns
- Block latency is 3581 clock cycle

3.4.6. Simulation Results:

3.4.6.1. MATLAB Results:

```
Columns 1 through 5

-1.1659 + 3.0525i   2.3489 + 1.6738i   3.0407 + 1.3423i   2.7699 + 1.5465i  -2.7852 - 1.3501i
-2.6331 - 1.3461i   2.6202 + 1.6800i  -2.9680 - 1.7880i   3.0301 + 1.4585i   1.6576 - 2.8561i
-2.7971 - 1.1928i   2.8437 + 1.5294i   3.3597 + 0.9794i  -1.3607 + 3.2405i  -3.0923 - 1.0654i
 1.4179 - 3.3584i  -1.3332 + 3.6711i  -0.9617 + 3.0577i   0.8886 - 3.3672i   3.5745 + 1.1439i
-2.8242 - 0.7913i  -3.5238 - 0.3251i  -0.7558 + 3.0005i  -3.3293 - 0.7405i   0.9789 - 3.1553i
-0.1751 + 2.9210i   0.3313 - 2.6833i   2.9801 + 0.3348i  -3.1303 - 0.5543i  -2.8261 - 0.5140i
-0.1929 + 2.6839i   2.7065 + 0.7988i  -2.9278 - 0.6070i   0.4312 - 2.8397i   0.6946 - 2.8428i
 2.7727 + 0.3935i  -2.6676 - 0.2357i   0.4337 - 2.6832i   0.5061 - 2.6149i  -2.4400 - 0.5709i
 2.5424 + 0.6152i   0.6227 - 2.2313i  -0.7346 + 2.4039i   2.4310 + 0.6612i  -0.5774 + 2.3940i
 0.9745 - 2.3097i   2.2897 + 1.0020i  -2.3569 - 0.8878i   2.3657 + 0.8613i   2.4240 + 0.8877i
-2.2867 - 0.9896i   2.2782 + 1.1579i  -1.0652 + 2.4267i  -2.3861 - 1.2201i   2.4191 + 1.1589i
 1.4451 - 2.3009i   2.3214 + 1.2686i  -2.3601 - 1.2919i  -2.5183 - 1.4253i  -2.5389 - 1.2857i

Columns 6 through 10

 0.9447 - 2.2370i  -0.0797 + 3.5405i  -3.8610 - 0.0410i  -2.5545 - 1.4066i   2.4354 + 1.6444i
-3.3709 - 1.7128i   1.3626 - 3.0943i   1.8378 + 1.0604i  -2.7944 - 1.3783i   1.1412 - 3.0364i
-1.5821 + 3.1715i  -2.8694 - 0.3863i  -1.7865 + 1.4808i   3.6309 + 1.4336i   2.9130 + 1.4958i
 0.6941 - 3.3942i   0.3608 - 2.9329i  -1.5010 - 2.8843i  -3.0225 - 0.9635i   3.0106 + 1.2670i
 2.9584 + 0.7426i  -3.1381 - 0.5723i  -1.5625 - 2.5413i  -0.4720 + 3.2644i   0.5107 - 2.9929i
-0.8219 + 3.1158i   0.3080 + 2.9591i   3.2352 - 2.5387i   0.8363 - 3.1091i  -3.2694 - 0.3263i
-2.2184 - 0.2788i  -0.2853 - 2.6946i   3.9101 - 1.9176i  -2.6590 - 0.5660i  -2.9686 - 0.2891i
 2.4549 + 0.6245i  -0.0519 - 3.2599i  -0.0928 - 0.7158i   2.7455 + 0.5423i  -2.6868 - 0.3955i
-0.8092 + 2.4956i  -2.6378 - 0.7254i  -1.1010 - 0.3540i   2.5096 + 0.7366i   2.4092 + 0.8604i
-2.4252 - 0.9093i   1.8225 - 0.4976i   1.8356 - 0.2601i   0.9727 - 2.2943i   2.3263 + 1.0074i
 2.3626 + 1.1266i  -2.4562 - 1.0689i   1.7442 - 0.5975i  -2.4017 - 1.0251i  -1.2720 + 2.3785i
 2.4917 + 1.2462i  -0.6488 + 2.4593i   0.9493 + 3.2896i   1.3835 - 2.2695i   2.4012 + 1.2721i
```

Figure 94 resource element de-mapper MATLAB results

3.4.6.2. RTL Results:

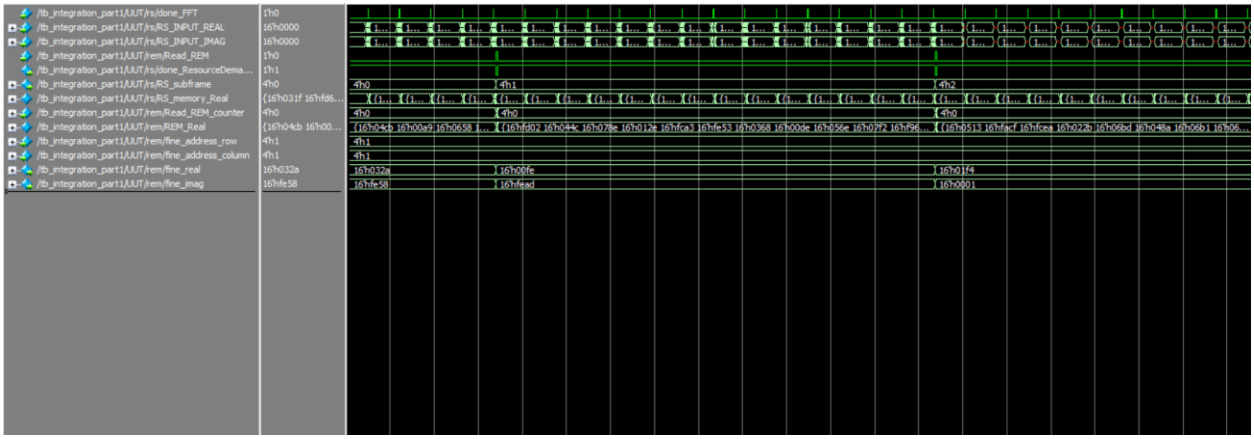


Figure 95 resource element de-mapper RTL simulation

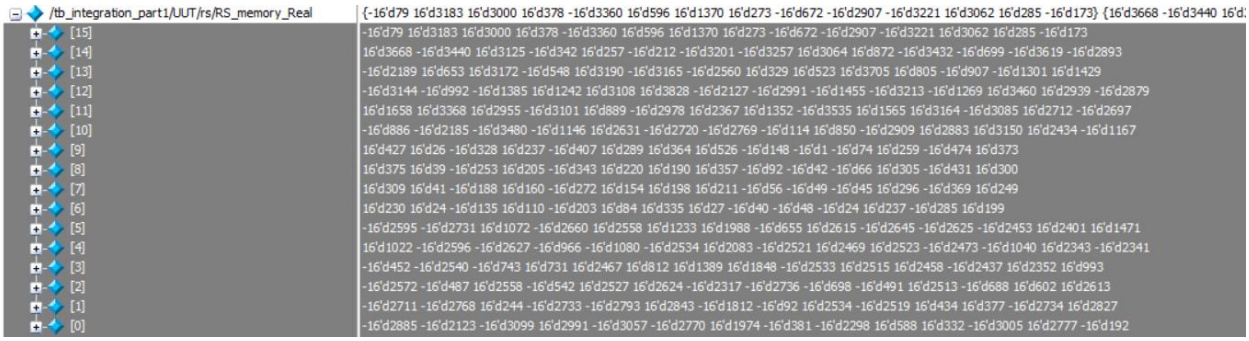


Figure 96 Storage element memory

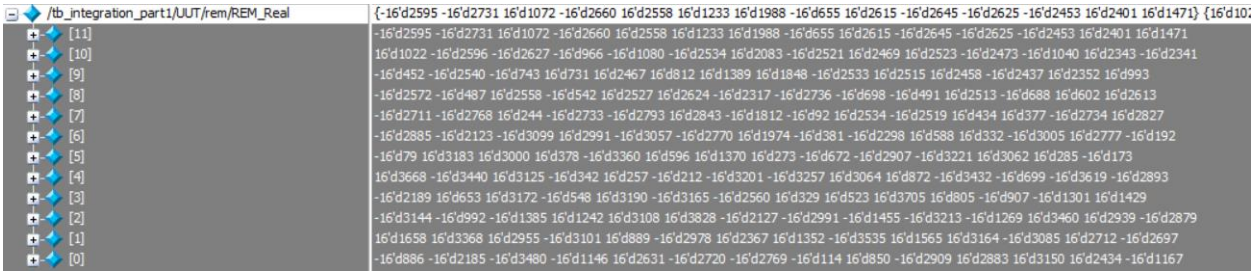


Figure 97 Resource De-mapper memory

3.4.6.3. Synthesis Reports:

Screenshots for synthesis results

```
Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  52.7699 uW   (98%)
  Net Switching Power  =   1.1581 uW    (2%)
                         ---------
Total Dynamic Power    =  53.9280 uW  (100%)

Cell Leakage Power     = 193.6498 uW


                 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          52.5237        1.3857e-02        9.7865e+04         150.4035  (  60.75%)
sequential         0.0000           0.0000            0.0000           0.0000  (   0.00%)
combinational      0.2441           1.1443        9.5775e+04           97.1636  (  39.25%)
--------------------------------------------------------------------------------------------------
Total             52.7678 uW        1.1582 uW     1.9364e+05 nW       247.5671 uW
1
```

Figure 98 Storage Element Power Report

```
Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  32.6744 uW   (85%)
  Net Switching Power  =   5.7121 uW   (15%)
                         ---------
Total Dynamic Power    =  38.3865 uW  (100%)

Cell Leakage Power     = 270.6989 uW


                 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          28.0336           0.3398        8.9822e+04          118.1950  (  38.24%)
sequential         0.0000           0.0000            0.0000            0.0000  (   0.00%)
combinational      4.6408           5.3722        1.8088e+05          190.8925  (  61.76%)
--------------------------------------------------------------------------------------------------
Total             32.6744 uW        5.7121 uW     2.7070e+05 nW       309.0876 uW
1
```

Figure 99 Resource element De-mapper Power Report

```
Number of ports:                        431
Number of nets:                       23081
Number of cells:                      21101
Number of combinational cells:        15316
Number of sequential cells:            5785
Number of macros:                         0
Number of buf/inv:                     6558
Number of references:                    27

Combinational area:       16930.102155
Buf/Inv area:             4775.231903
Noncombinational area:    26173.335057
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:          43103.437212
Total area:                 undefined
1
```

Figure 100 Storage Element Area Report

```
Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  32.6744 uW   (85%)
  Net Switching Power  =   5.7121 uW   (15%)
                         ---------
Total Dynamic Power    =  38.3865 uW  (100%)

Cell Leakage Power     = 270.6989 uW
```

Figure 101 Resource Element De-Mapper Area Report

```
clock clk (rise edge)                                  520.00     520.00
clock network delay (ideal)                              0.00     520.00
clock uncertainty                                       -0.35     519.65
RS_memory_Imag_reg[15][3][12]/CK (DFF_X1)                0.00     519.65 r
library setup time                                      -0.04     519.61
data required time                                                519.61
-------------------------------------------------------------------------
data required time                                                519.61
data arrival time                                                  -1.76
-------------------------------------------------------------------------
slack (MET)                                                       517.85
```

Figure 102 Storage Element Timing Report

```
clock clk (rise edge)                                  520.00     520.00
clock network delay (ideal)                              0.00     520.00
clock uncertainty                                       -0.35     519.65
output external delay                                   -1.00     518.65
data required time                                                518.65
-------------------------------------------------------------------------
data required time                                                518.65
data arrival time                                                  -1.64
-------------------------------------------------------------------------
slack (MET)                                                       517.01
```

Figure 103 Resource Element De-Mapper Timing Report

3.5. Channel Estimation

3.5.1. Block Diagram:

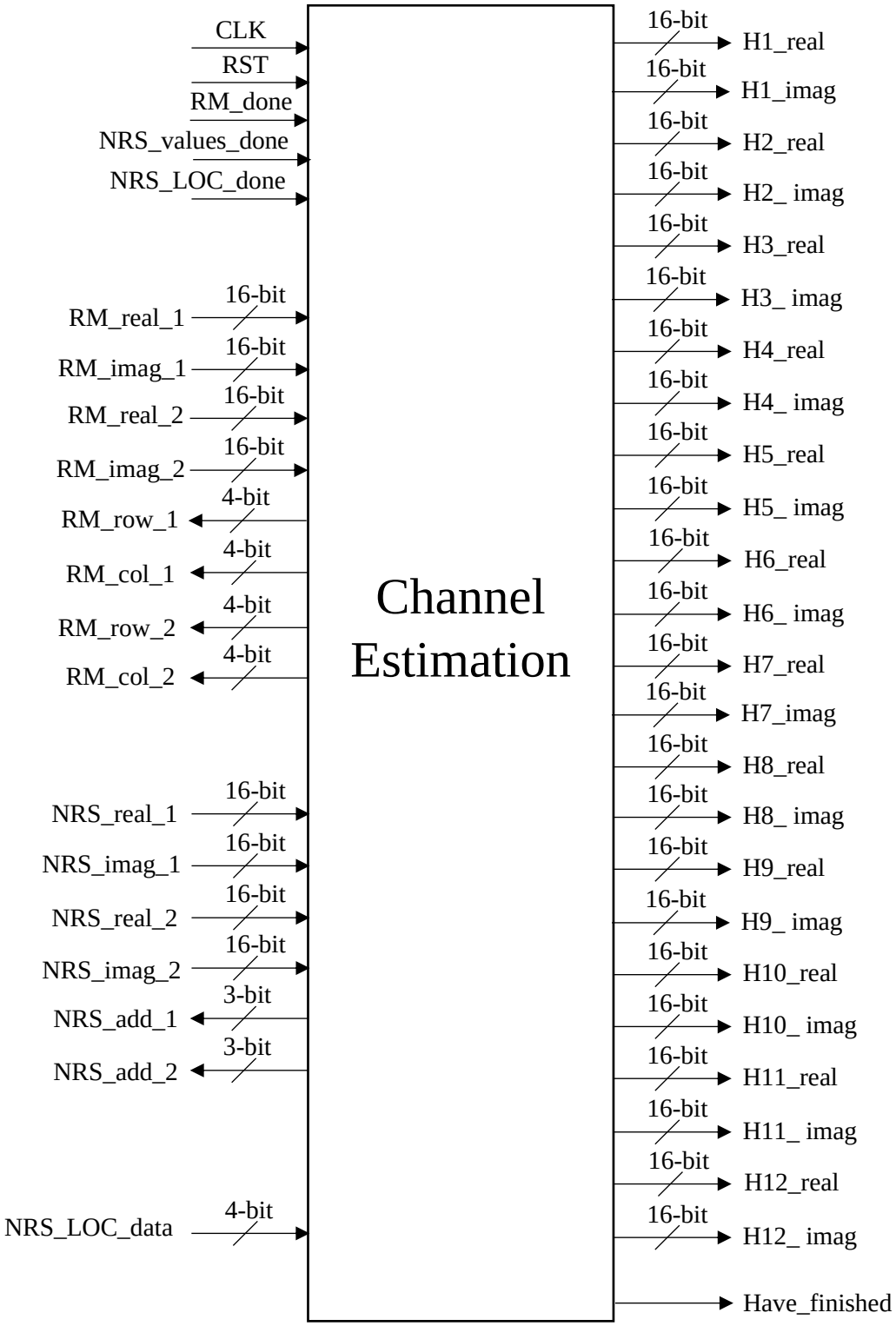


Figure 104 Channel Estimation Block Diagram

3.5.2. Interface Table:

Table 18 Channel Estimation interface table

Signal Name	Direction	Width	Description
CLK	Input	1-bit	Clock of the system 520 ns
RST	Input	1-bit	Reset of the system
RM_done	Input	1-bit	Output Resource Mapper enable to be high after symbols storing finished
NRS_values_done	Input	1-bit	Output NRS values enable to be high after NRS values storing finished
NRS_LOC_done	Input	1-bit	Output NRS indices enable to be high after NRS indices storing finished
RM_real_1	Input	16-bits	Received real value pilots of the 1st slot
RM_imag_1	Input	16-bits	Received imaginary value pilots of the 1st slot
RM_real_2	Input	16-bits	Received real value pilots of the 2nd slot
RM_imag_2	Input	16-bits	Received imaginary value pilots of the 2nd slot
NRS_real_1	Input	16-bits	Transmitted real value pilots of the 1st slot
NRS_imag_1	Input	16-bits	Transmitted imaginary value pilots of the 1st slot
NRS_real_2	Input	16-bits	Transmitted real value pilots of the 2nd slot
NRS_imag_2	Input	16-bits	Transmitted imaginary value pilots of the 2nd slot
NRS_LOC_data	Input	4-bits	Indices values of the received pilots
RM_row_1	Output	4-bits	1st Address of the rows to get the data from the Resource Mapper
RM_col_1	Output	4-bits	1st Address of the columns to get the data from the Resource Mapper
RM_row_2	Output	4-bits	2nd Address of the rows to get the data from the Resource Mapper
RM_col_2	Output	4-bits	2nd Address of the columns to get the data from the Resource Mapper
NRS_add_1	Output	3-bits	Address for the Transmitted/Generated Pilots of the 1st slot and the indices values.
NRS_add_2	Output	3-bits	Address for the Transmitted/Generated Pilots of the 2nd slot.
H1_real	Output	16-bits	1st sub-carrier real channel frequency response(H_{LS}).
H1_imag	Output	16-bits	1st sub-carrier imaginary channel frequency response(H_{LS}).
H2_real	Output	16-bits	2nd sub-carrier real channel frequency response(H_{LS}).
H2_imag	Output	16-bits	2nd sub-carrier imaginary channel frequency response(H_{LS}).
H3_real	Output	16-bits	3rd sub-carrier real channel frequency response(H_{LS}).
H3_imag	Output	16-bits	3rd sub-carrier imaginary channel frequency response(H_{LS}).
H4_real	Output	16-bits	4th sub-carrier real channel frequency response(H_{LS}).
H4_imag	Output	16-bits	4th sub-carrier imaginary channel frequency response(H_{LS}).
H5_real	Output	16-bits	5th sub-carrier real channel frequency response(H_{LS}).
H5_imag	Output	16-bits	5th sub-carrier imaginary channel frequency response(H_{LS}).
H6_real	Output	16-bits	6th sub-carrier real channel frequency response(H_{LS}).
H6_imag	Output	16-bits	6th sub-carrier imaginary channel frequency response(H_{LS}).
H7_real	Output	16-bits	7th sub-carrier real channel frequency response(H_{LS}).
H7_imag	Output	16-bits	7th sub-carrier imaginary channel frequency response(H_{LS}).

H8_real	Output	16-bits	8th sub-carrier real channel frequency response(H_{LS}).
H8_imag	Output	16-bits	8th sub-carrier imaginary channel frequency response(H_{LS}).
H9_real	Output	16-bits	9th sub-carrier real channel frequency response(H_{LS}).
H9_imag	Output	16-bits	9th sub-carrier imaginary channel frequency response(H_{LS}).
H10_real	Output	16-bits	10th sub-carrier real channel frequency response(H_{LS}).
H10_imag	Output	16-bits	10th sub-carrier imaginary channel frequency response(H_{LS}).
H11_real	Output	16-bits	11th sub-carrier real channel frequency response(H_{LS}).
H11_imag	Output	16-bits	11th sub-carrier imaginary channel frequency response(H_{LS}).
H12_real	Output	16-bits	12th sub-carrier real channel frequency response(H_{LS}).
H12_imag	Output	16-bits	12th sub-carrier imaginary channel frequency response(H_{LS}).
Have_finished	Output	1-bit	Output Enable from the channel estimation to indicate that calculation of all channel frequency responses finished

3.5.3. Function of the design:

Channel estimation process take several stages to get the channel frequency response to the equalizer to compensate the effect of the channel:

1. Get the Transmitted/Generated pilots out from the NRS generator.
2. Get the Received pilots out from the Resource Mapper.
3. Calculate the channel frequency response.
4. Store the four channel frequency responses got from the eight pilots.
5. Interpolate the output to get finally the twelve channel frequency responses to the equalizer.

3.5.4. Block Specification:

- Rate: the clock used is 520 ns
- Latency: the design is pipeline to use the same hardware with low latency so it uses 7 clock cycles to get the finish enable high.

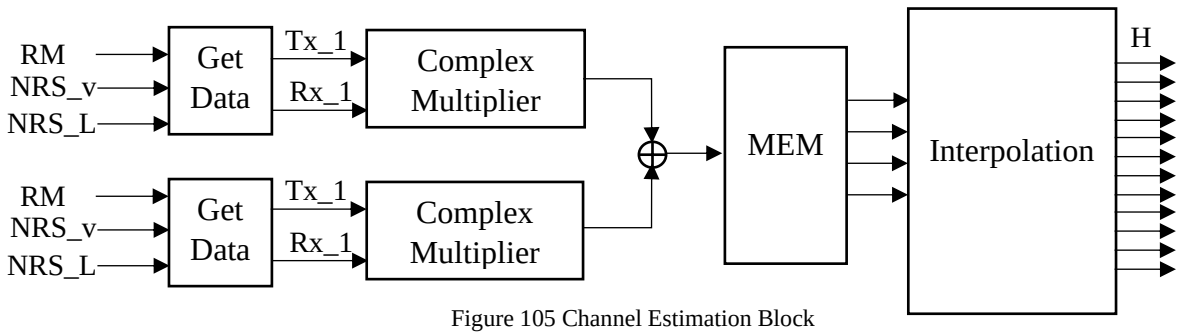


Figure 105 Channel Estimation Block diagram

3.5.5. Detailed Design Implementation:

A. Get Data from the RM and the NRS generator:

The purpose of this block is to get the data synchronously from the Resource Mapper and the NRA generator to provide the transmitted and received pilots to the complex multiplier.

Timing is very important in this block, as each pilot taken to the first path should be equal in the sub carrier but differ in the slot number from that taken to the second path.

So that we can take the average of the right pilots.

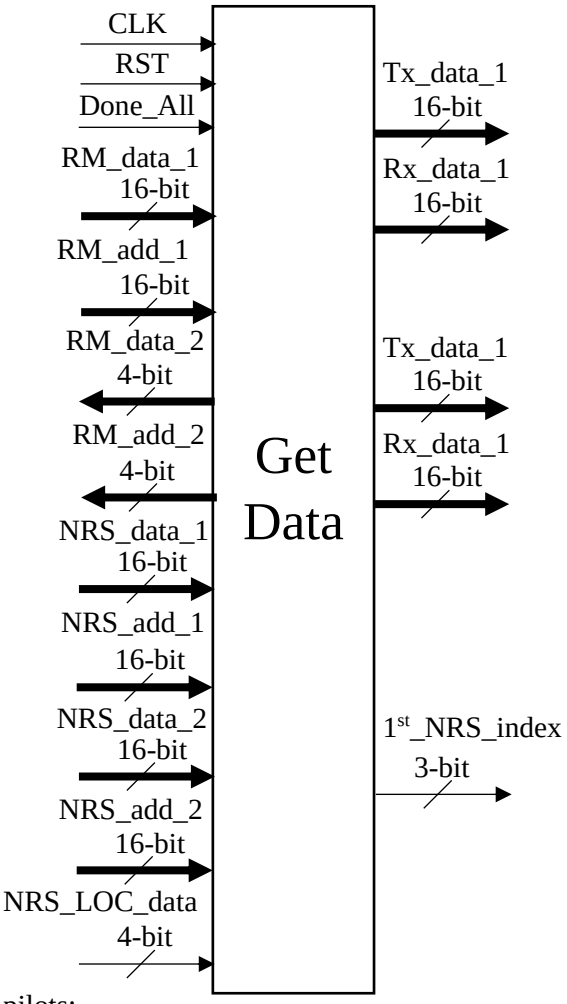


Figure 106 Channel Estimation-Get Data Block Diagram

B. Calculate the channel frequency response using pilots:

This stage is corresponding to the divider ($x^{-1}y$) but to optimize in area and hardware so we benefit from the idea of that the value of any pilot is $1/\sqrt{2}$ whatever the sign is.

Hence, conjugate will transfer the division operation into multiplication operation and the shift lift operation divide the value by two.

Then the output from the two paths added to take the average.

$$\left(\pm\frac{1}{\sqrt{2}}\pm\frac{1}{\sqrt{2}}i\right)\left(\pm\frac{1}{\sqrt{2}}\mp\frac{1}{\sqrt{2}}i\right)=1$$

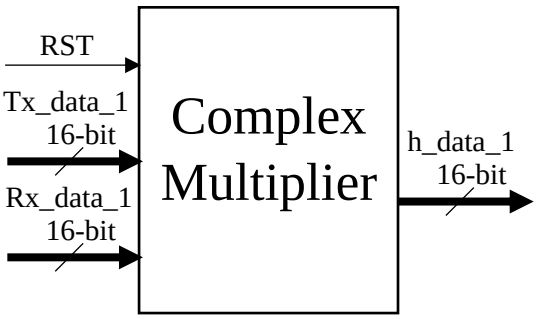


Figure 107 Channel Estimation-Complex Multiplier Block Diagram

C. Register file to store the four channels frequency response:

Sequential block to store the values of the channel frequency response for the four sub-carriers that contain the eight pilots.

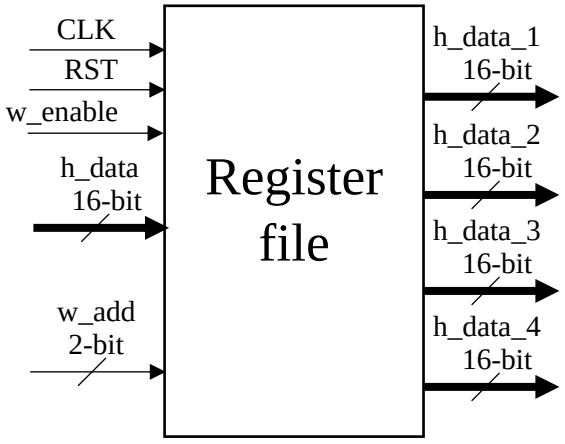


Figure 108 Channel Estimation-Register file Block Diagram

D. Interpolation:

Combinational block and its function are to calculate the linear interpolation of the four pilots to get the twelve channel frequency responses.

The idea based on two MUXs one with 2-bits selection and the other with 1-bit selection to get the 6 possible probabilities according to the different of the location of each input channel frequency response as it depend on the N_{ID}^{cell}.

So, the input "index_1st_NRS" come from the NRS Location generator used as the selection of the two MUXs according to the Location of the first pilot that change when N_{ID}^{cell} change.

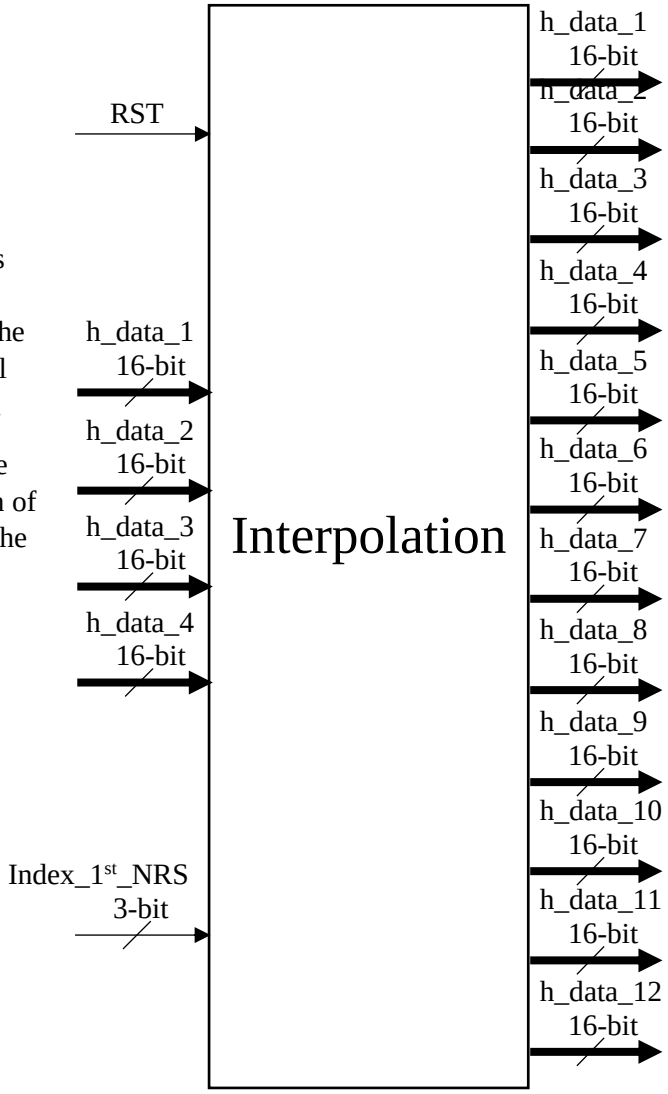


Figure 109 Channel Estimation-Interpolation Block Diagram

3.5.6. Design Interface:

Channel Estimation block communicate with four blocks:

Provide the input data from NRS values generator, NRS Location generator, and the Resource mapper

Support its output channel frequency response data to the channel equalizer

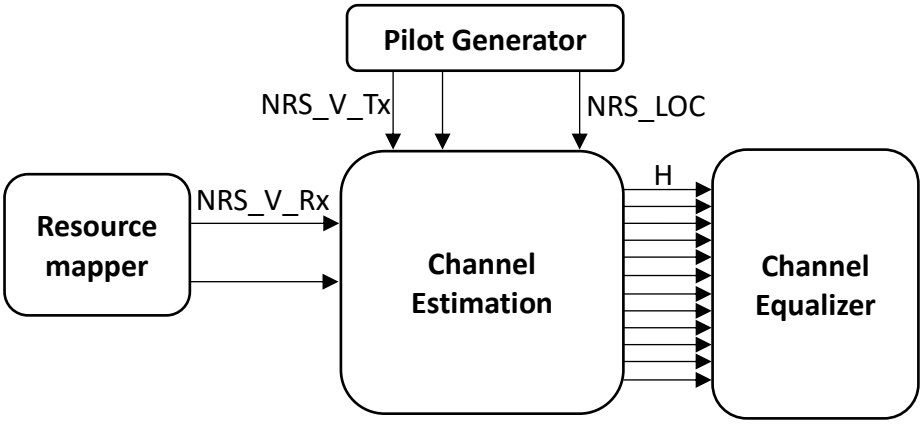


Figure 110 Channel Estimation interface block diagram

3.5.7. Simulation Results:

3.5.7.1. MATLAB Results:

The performance of channel estimation calculated with the BER.

So, the BER < 10% for SNR > 3 dB.

Test case depend on take BER average of 10 different input data repetitions for each SNR value.

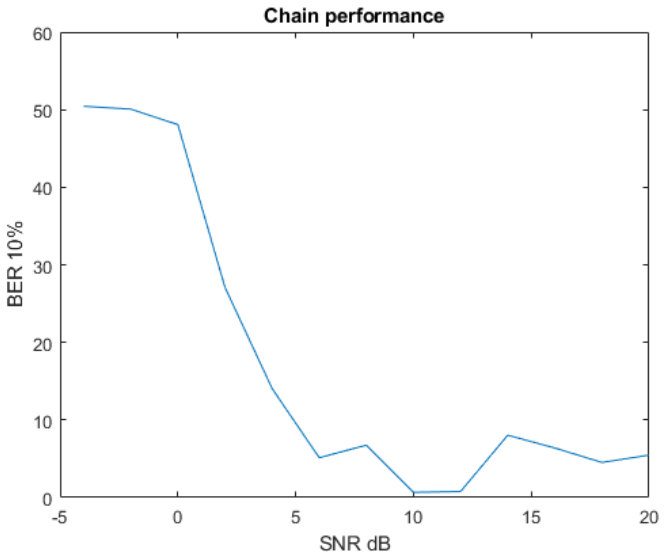


Figure 111 Channel Estimation performance

3.5.7.2. RTL Results:

The Input assumptions:

NRS_done = 1; Tx_real = 1,2,1,2 2,2,1,1
LOC_done = 1; NRS_LOC = 0,3,6,9 0,3,6,9
RM_done = 1; Rx_real = 0,2,4,6 0,1,2,3

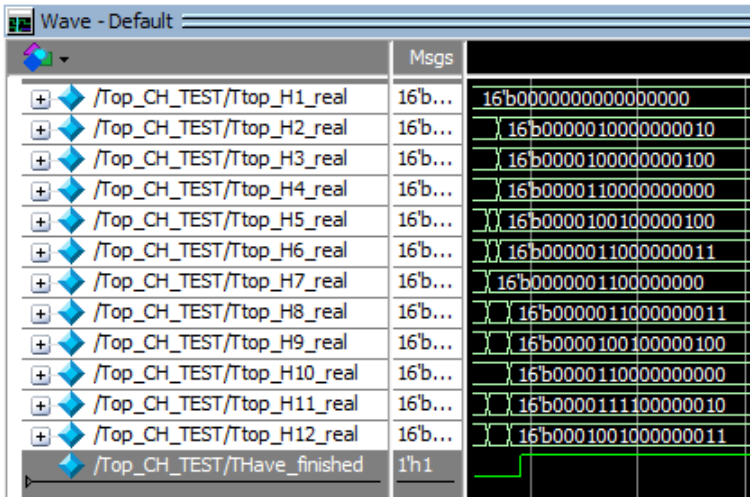


Figure 112 RTL Channel estimation real outputs with fixed-point approximation

Table 19 Channel Estimation interface table

Actual output:	Expected output:
0	0
1.001953125	1
2.00390625	2
3	3
2.25390625	2.25
1.5029296875	1.5
0.75	0.75
1.5029296875	1.5
2.25390625	2.25
3	3
3.751953125	3.75
4.502929688	4.5

3.5.7.3. Synthesis Reports:

```
Combinational area:       19017.138082
Buf/Inv area:               840.293999
Noncombinational area:      893.760029
Net Interconnect area:       undefined

Total cell area:          19910.898111
Total area:                  undefined
1
```

Figure 113 RTL Channel estimation area report

```
  Cell Internal Power  =  86.6037 uW   (70%)
  Net Switching Power  =  36.6190 uW   (30%)
                         ----------
Total Dynamic Power    = 123.2226 uW  (100%)

Cell Leakage Power     =  78.8232 uW
```

Figure 114 RTL Channel estimation power report

```
clock top_ch_CLK (rise edge)                          520.00     520.00
clock network delay (ideal)                             0.00     520.00
clock uncertainty                                      -0.35     519.65
top_ch_real_reg[11]/CK (DFFR_X1)                        0.00     519.65 r
library setup time                                     -0.03     519.62
data required time                                               519.62
-------------------------------------------------------------------------
data required time                                               519.62
data arrival time                                                 -5.04
-------------------------------------------------------------------------
slack (MET)                                                      514.58
```

Figure 115 RTL Channel estimation timing report

3.6. NRS Value Generation

3.6.1. Block Diagram:

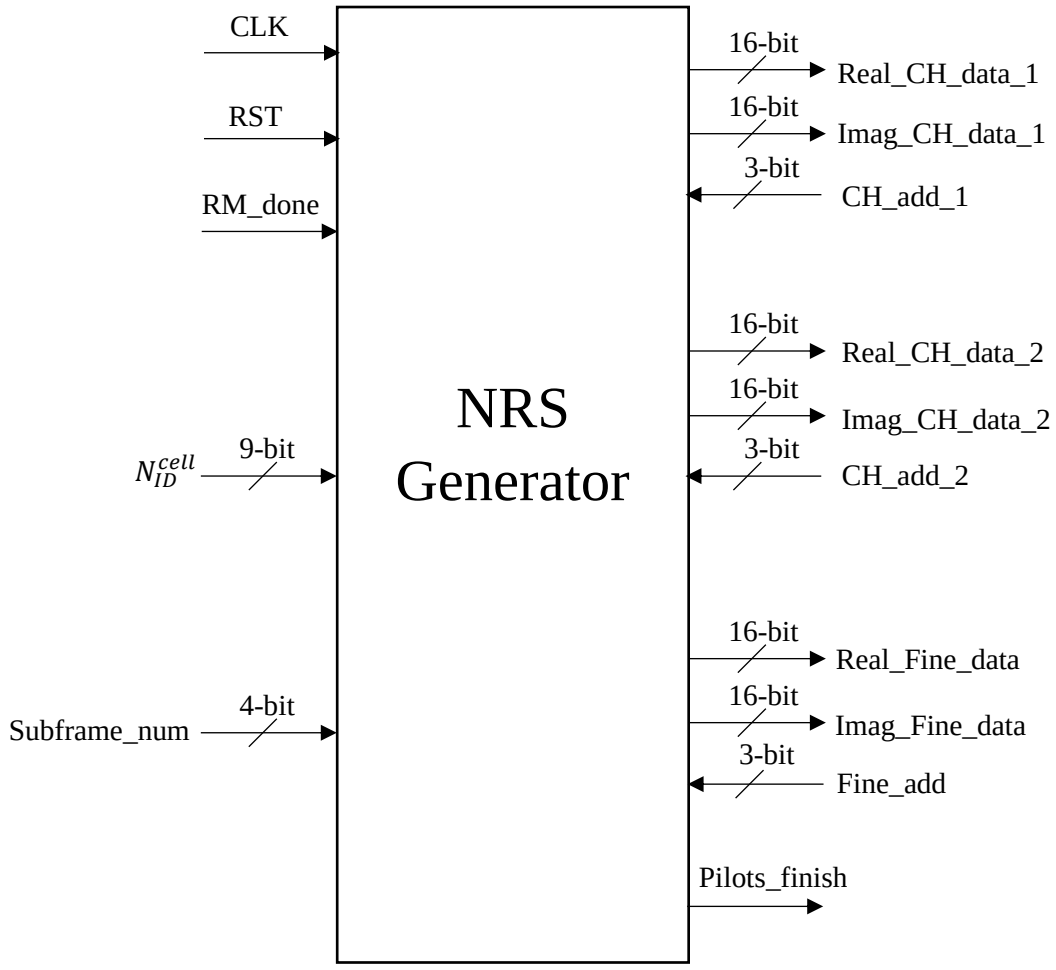


Figure 116 NRS Generator Block Diagram

3.6.2. Interface Table:

Table 20 NRS generator interface table

Signal Name	Direction	Width	Description
CLK	Input	1-bit	The clock for the synchronous blocks should be fast clock due to the very high latency
RST	Input	1-bit	To the reset the block
RM_done	Input	1-bit	To start and restart when the resource mapper finish storing data
N_{ID}^{cell}	Input	9-bits	Input to get the NRS Location
Subframe_num	Input	4-bits	Number of the sub frame ranged from 0 to 9
CH_add_1	Input	3-bits	Address for 1st path of the Channel estimation that read NRS

CH_add_2	Input	3-bits	Address for 2nd path of the Channel Estimation that read NRS
Fine_add	Input	3-bits	Address for the Fine Synchronization block that read NRS
Real_CH_data_1	Output	16-bits	Real NRS data output from the pilot generation to the Channel Estimation 1st path
Imag_CH_data_1	Output	16-bits	Imaginary NRS data output from the pilot generation to the Channel Estimation 1st path
Real_CH_data_2	Output	16-bits	Real NRS data output from the pilot generation to the Channel Estimation 2nd path
Imag_CH_data_2	Output	16-bits	Imaginary NRS data output from the pilot generation to the Channel Estimation 2nd path
Real_Fine_data	Output	16-bits	Real NRS data output from the pilot generation to the Fine Synchronization
Imag_Fine_data	Output	16-bits	Imaginary NRS data output from the pilot generation to the Fine Synchronization
Pilots_finish	Output	1-bit	The output to indicate the finish of the pilot generation process

3.6.3. Function of the design:

The NRS Generation Block aim to generate the real and imaginary values of the narrowband reference signals and store them in a right order inside a register file to be available.

The sequence of generating NRS real and imaginary values is:

1. Get the second m-sequence generator.
2. Use Lift Feedback Shift Register to get the required sequence.
3. Get the values of the pilots according to the fixed point and the sequence result "16'b111111_01_00101011 / 16'b000000_1011010101".
4. Store the final values inside a register file in a right order.

3.6.4. Block Specification:

- Rate: the clock used is 130 ns
- Latency: 6424 clock cycle.

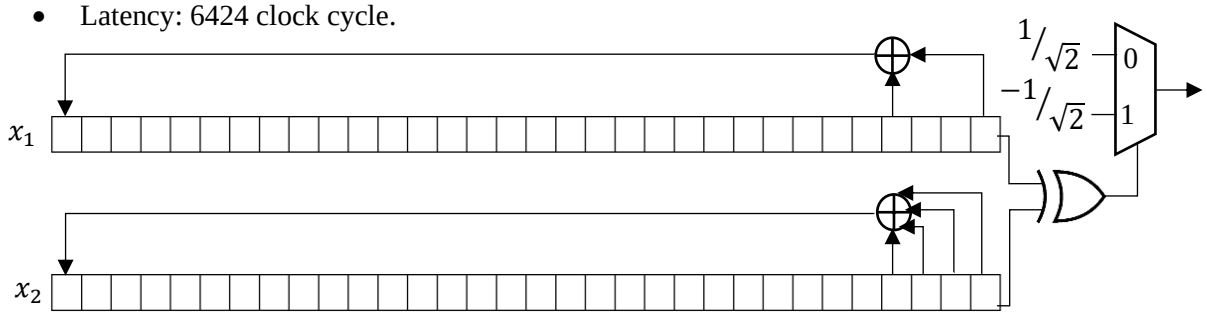


Figure 117 NRS values generator Block diagram

3.6.5. Detailed Design Implementation:

A. C_{init} initializing of second m-sequence:

It is a combinational block used to get the initial value of the second m-sequence

With an assumption that N_{ID}^{cell} dose not change in the middle of the data transmission, so there are no extra hardware to check the variations of the N_{ID}^{cell} and whenever N_{ID}^{cell} changes the Init_out will changes immediately.

Also, for the number of the sub-frame as the number cannot change in the middle of a sub-frame so no hardware check for its variations and whenever sub-frame number changes the Init_out changes immediately.

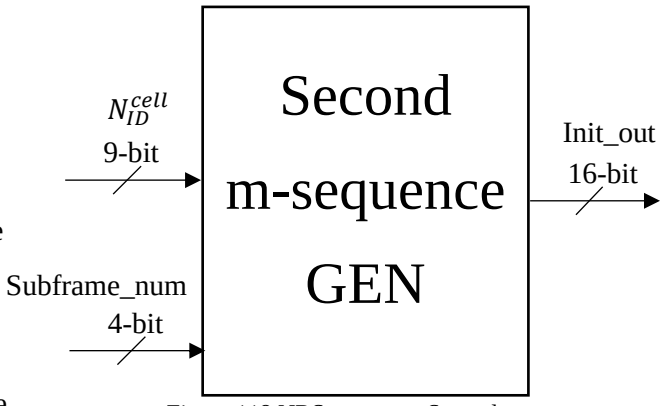


Figure 118 NRS generator Second m-sequence Generator block diagram

B. LFSR with parallel input to generate the pilots for each sub frame:

Depends mainly on two counters to calculate the pilots with least area and power and use fast clock to compensate the large delay.

The enables and address used to store the pilots inside the register file well and the finish indicate that the storing process finished.

As a general goal to get the pilots, I use only two LFSR to generate NRS that may be optimal.

As for case of only one LFSR I will use a temporarily memory that mean extra hard ware and the latency will be very large that make me use a very fast clock then more power.

Also, for use more than two LFSR that mean very low latency but extra huge area and power.

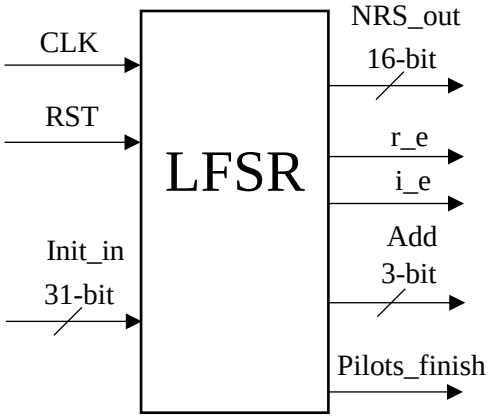


Figure 119 NRS Generator LFSR Block diagram

C. Register file to store the pilots during the sub frame to be used by the channel estimation and fine Synchronization blocks:

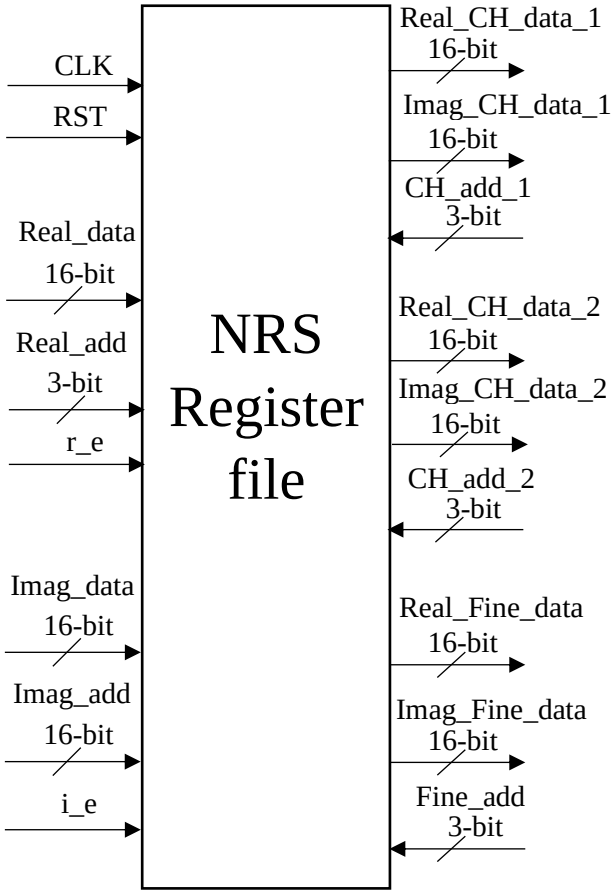


Figure 120 NRS generator Register files block diagram

3.6.6. Design Interface:

Three Blocks must communicate with the NRS Generator:
Provide sub-frame number from the synchronization block path by FFT block.
Support NRS output data to the Channel Estimator block for using them to estimate the channel.
Support NRS output data to the Fine Synchronization block for using in a synchronization algorism.

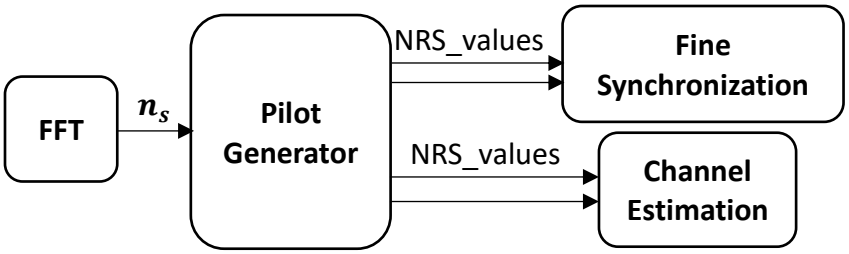
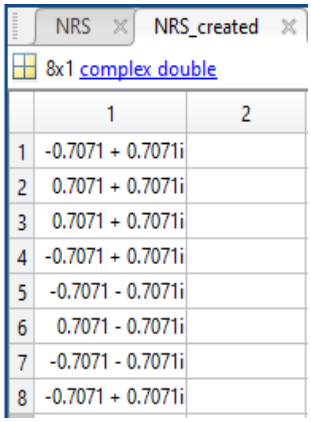


Figure 121 NRS values generator interface Block diagram

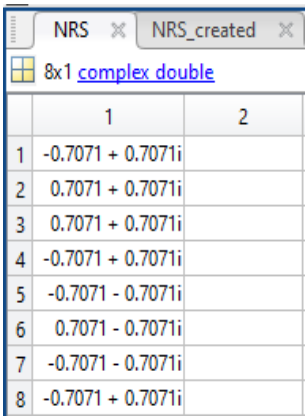
3.6.7. Simulation Results:

3.6.7.1. MATLAB Results:



	1	2
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 123 NRS Values generated by created MATLAB function for sub frame number = 6



	1	2
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 122 NRS Values generated by built in MATLAB function for sub frame number = 6

3.6.7.2. RTL Results:

- Assuming inputs: $N_{ID}^{cell} = 1$, Sub Frame number = 0

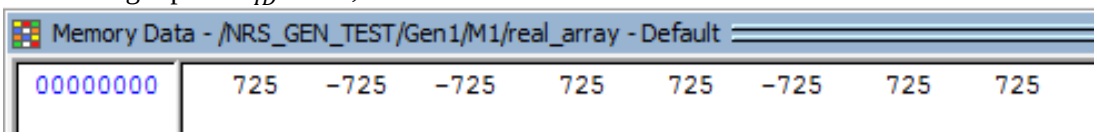


Figure 124 NRS Values generated by RTL block

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

Figure 125 NRS Values generated from the MATLAB

3.6.7.3. Synthesis Reports:

```
  Cell Internal Power  =  95.7486 uW   (89%)
  Net Switching Power  =  11.8745 uW   (11%)
                         ---------
Total Dynamic Power    = 107.6232 uW  (100%)

Cell Leakage Power     =  15.3137 uW
```

Figure 126 NRS Values generation block power report

```
Combinational area:       1672.874010
Buf/Inv area:              127.945999
Noncombinational area:    1766.240057
Net Interconnect area:      undefined

Total cell area:          3439.114067
```

Figure 127 NRS Values generation block area report

```
clock CLK (rise edge)                   130.00     130.00
clock network delay (ideal)               0.00     130.00
clock uncertainty                        -0.35     129.65
output external delay                    -2.00     127.65
data required time                                 127.65
-----------------------------------------------------------
data required time                                 127.65
data arrival time                                   -2.32
-----------------------------------------------------------
slack (MET)                                        125.33
```

Figure 128 NRS Values generation block timing report

3.7. NRS Index Generation

3.7.1. Block Diagram:

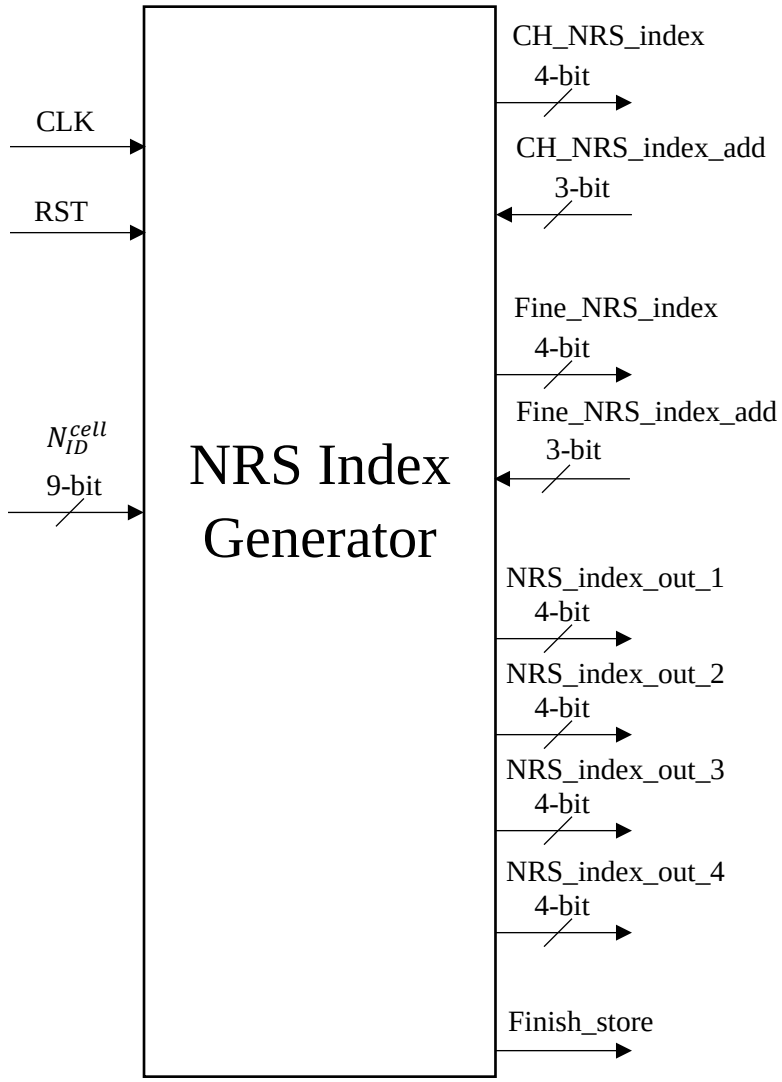


Figure 129 NRS Index Generator Block Diagram

3.7.2. Interface Table:

Table 21 NRS Index Generator interface table

Signal Name	Direction	Width	Description
CLK	Input	1-bit	The Clock of the system.
RST	Input	1-bit	The reset of the system.
N_{ID}^{cell}	Input	9-bits	$N_{ID}^{cell}$is an input to the block.
CH_NRS_index_add	Input	3-bits	The address the Channel Estimation used to get the indices values.
Fine_NRS_index_add	Input	3-bits	The address the Fine Synchronization used to get the indices values.
CH_NRS_index	Output	4-bits	The indices values out to the Channel Estimation.

Fine_NRS_index	Output	4-bits	The indices values out to the Fine Synchronization.
NRS_index_out_1	Output	4-bits	The 1st index value out to the NRS Removal.
NRS_index_out_2	Output	4-bits	The 2nd index value out to the NRS Removal.
NRS_index_out_3	Output	4-bits	The 3rd index value out to the NRS Removal.
NRS_index_out_4	Output	4-bits	The 4th index value out to the NRS Removal.
Finish_store	Output	1-bit	The done flag indicate that the Generation finished

3.7.3. Function of the design:

The NRS Location Generator block aim to generate the indices values of the pilot according to the variation of the N_{ID}^{cell}.

The Sequence to generate the indices values is:

1. Get the N_{ID}^{cell} as an input and calculate $mode(N_{ID}^{cell}, 6)$ using counter to be synthesized operation.
2. Get the eight indices of the pilots for the current N_{ID}^{cell}.
3. Store the values inside a Register File.

3.7.4. Block Specification:

- Rate: the clock used is 130 ns
- Latency: 8/519 clock cycle varies according to the input N_{ID}^{cell}.

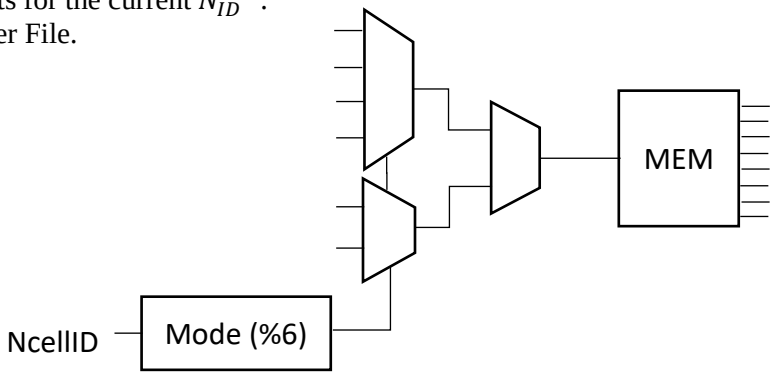


Figure 130 NRS Location generator Block diagram

3.7.5. Detailed Design Implementation:

1. Calculate modulus operation:

It is a sequential block based on a counter technique to get the modulus of six and use flag to re-calculate it when N_{ID}^{cell} change.

However, according to the assumption of constant N_{ID}^{cell} which mean that it does not change in the middle of the data transmission so any change of it will make an immediate change in the output of the modules and no extra hardware used to check for the variations of N_{ID}^{cell}

2.

The "Done" signal of this block get out when the $mode(N_{ID}^{cell}, 6)$ finishes so an XOR used between the value of N_{ID}^{cell} and the counter to be one only when the counter reach the same value as the input N_{ID}^{cell}.

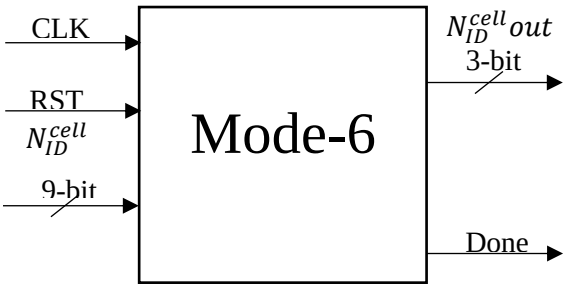


Figure 131 NRS index GEN mod-6 Block Diagram

1. Get each index corresponding to each pilot:

It is a sequential block to get the address of the places of the indices using counter.

The value of these indices calculated using the same counter as a selection for a MUX to get the value of the indices.

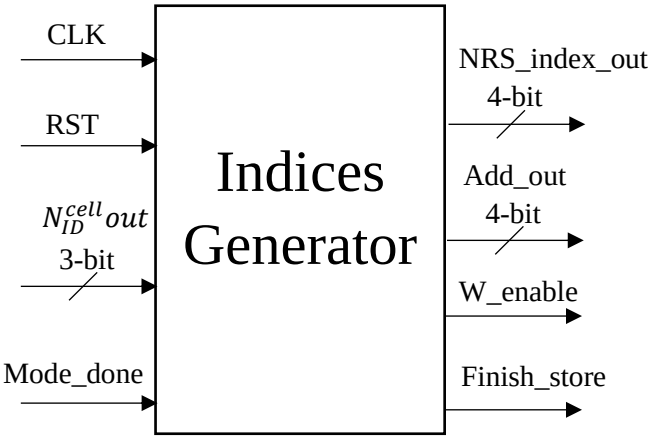


Figure 132 NRS index GEN indices Generator

2. Register file to store the eight indices for the eight pilots of the current N_{ID}^{cell}:

It stores the indices values so that Channel estimation, fine Synchronization and NRS Removal blocks could take them.

It is obvious that the bus to the channel estimation and fine synchronization is series while the bus to the NRS Removal is parallel.

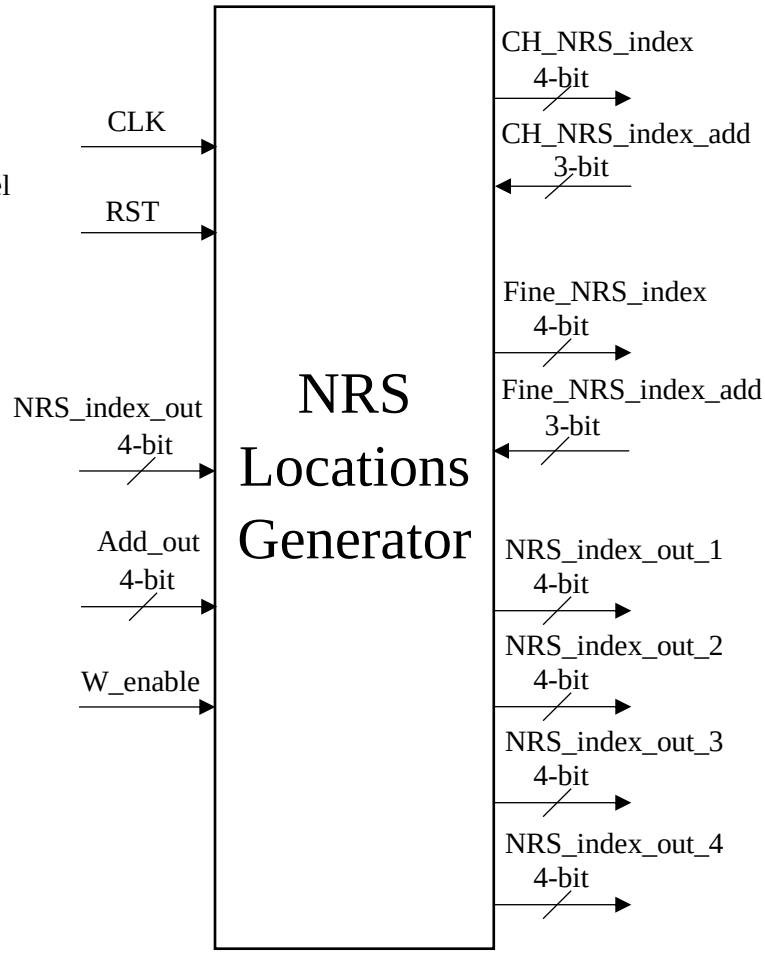


Figure 133 NRS Locations Generator

3.7.6. Design Interface:

Three Blocks must communicate with the NRS Location Generator:
Support NRS indices output data to the Channel Estimation used them to get the received pilots location.
Support NRS indices output data to fine synchronization used them to get the received pilots location.
Support NRS indices output data to the NRS Removal to be able to remove the NRS from their right locations.

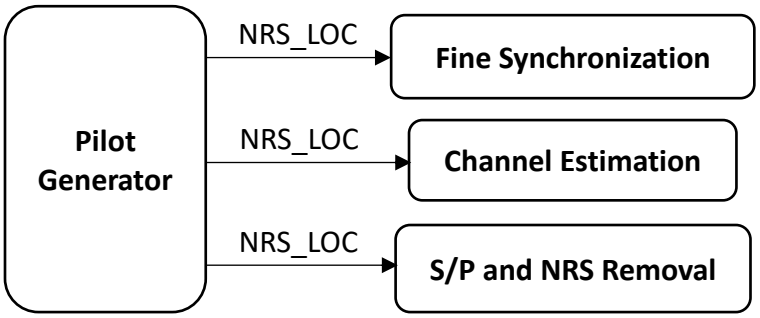


Figure 134 NRS Location generator Block diagram

3.7.7. Simulation Results:

3.7.7.1. MATLAB Results:

Assume the input of the Matlab created function is $N_{ID}^{cell} = 1$.
The expected output values of the eight pilots' indices is:

Table 22 MATLAB results for NRS index with $N_{ID}^{cell} = 1$

The pilots order	The value of index
1st pilot	1
2nd pilot	4
3rd pilot	7
4th pilot	10
5th pilot	1
6th pilot	4
7th pilot	7
8th pilot	10

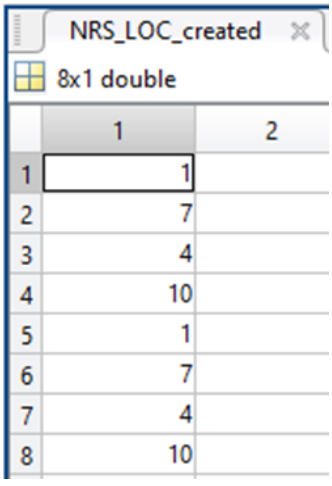


Figure 135 figure of the MATLAB function output for $N_{ID}^{cell} = 1$

3.7.7.2. RTL Results:

Assume the input of the Matlab created function is $N_{ID}^{cell} = 5$.
The expected output values of the eight pilots' indices is:

Table 23 MATLAB results for NRS index with $N_{ID}^{cell} = 5$

The pilots order	The value of index
1st pilot	5
2nd pilot	11
3rd pilot	2
4th pilot	8
5th pilot	5
6th pilot	11
7th pilot	2

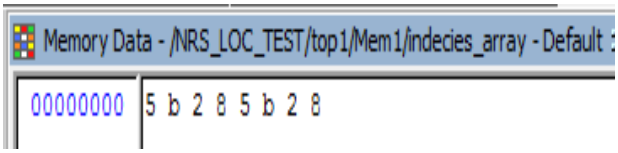


Figure 136 figure of the MODELSIM function output for $N_{ID}^{cell} = 5$

3.7.7.3. Synthesis Reports:

```
Combinational area:        240.464002
Buf/Inv area:               22.344000
Noncombinational area:     260.680008
Net Interconnect area:      undefined

Total cell area:           501.144011
```

Figure 138 NRS Location Generator block area report

```
  Cell Internal Power  =  17.4026 uW   (92%)
  Net Switching Power  =   1.6115 uW    (8%)
                         ---------
Total Dynamic Power    =  19.0141 uW  (100%)

Cell Leakage Power     =   2.4879 uW
```

Figure 137 NRS Location Generator block power report

```
clock top_CLK (rise edge)                              130.00     130.00
clock network delay (ideal)                              0.00     130.00
clock uncertainty                                       -0.35     129.65
Mem1/indecies_array_reg[7][2]/CK (DFFR_X1)               0.00     129.65 r
library setup time                                      -0.03     129.62
data required time                                                129.62
-------------------------------------------------------------------------
data required time                                                129.62
data arrival time                                                  -2.83
-------------------------------------------------------------------------
slack (MET)                                                       126.78
```

Figure 139 NRS Location Generator block timing report

3.8. Channel Equalizer

3.8.1. Block Diagram:

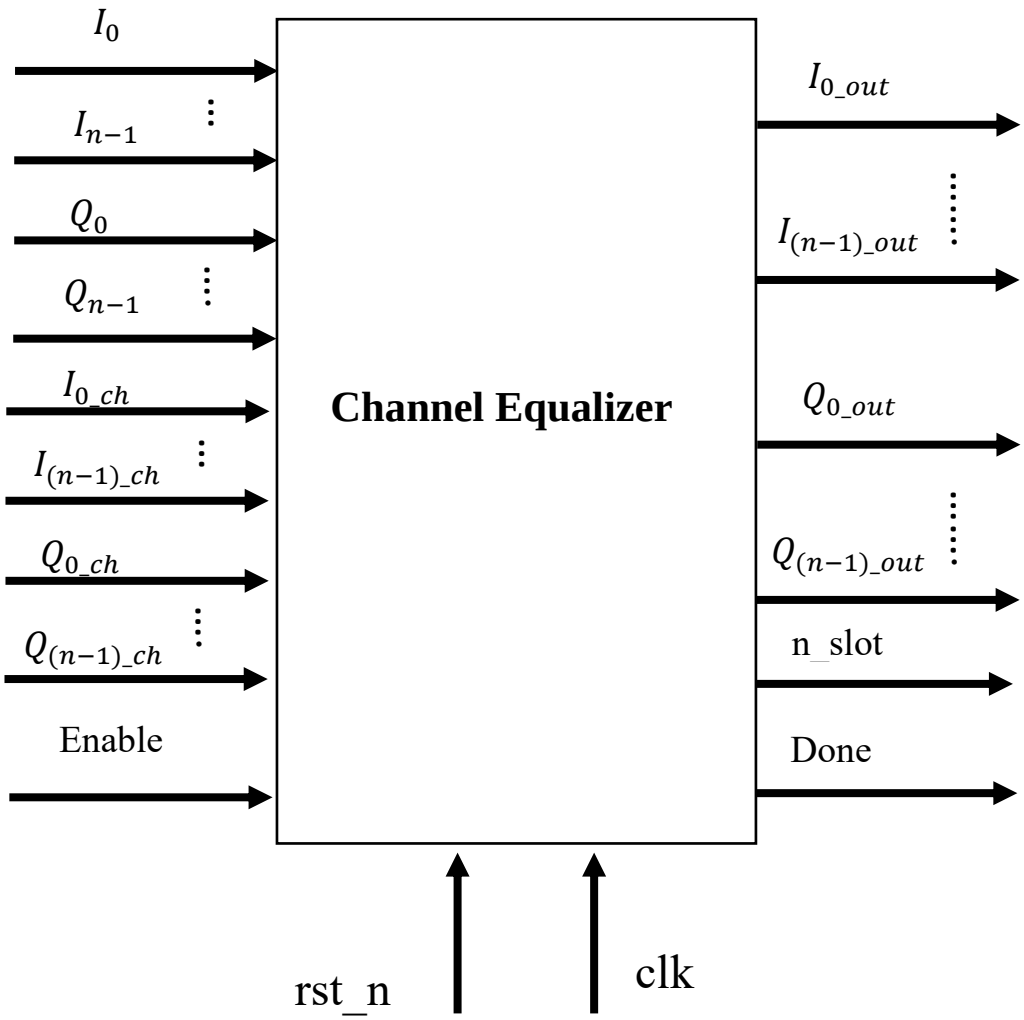


Figure 140 : Channel equalizer block diagram

3.8.2. Interface Table:

Table 24: interface table of channel equalizer

Signal Name	Direction	Width	Description
I_n	Input	16	Symbol real part
Q_n	Input	16	Symbol imaginary part
I_{n_ch}	Input	16	Estimated channel real part
Q_{n_ch}	Input	16	Estimated channel imaginary part
Enable	Input	1	Equalizer enable
clk	Input	1	Equalizer clock
Rst_n	Input	1	Equalizer reset

I_{n_out}	Output	16	Equalized symbol real part
Q_{n_out}	Output	16	Equalized symbol imaginary part
Done	Output	1	Signal indicate that equalizer is done
n_slot	Output	4	Slot number

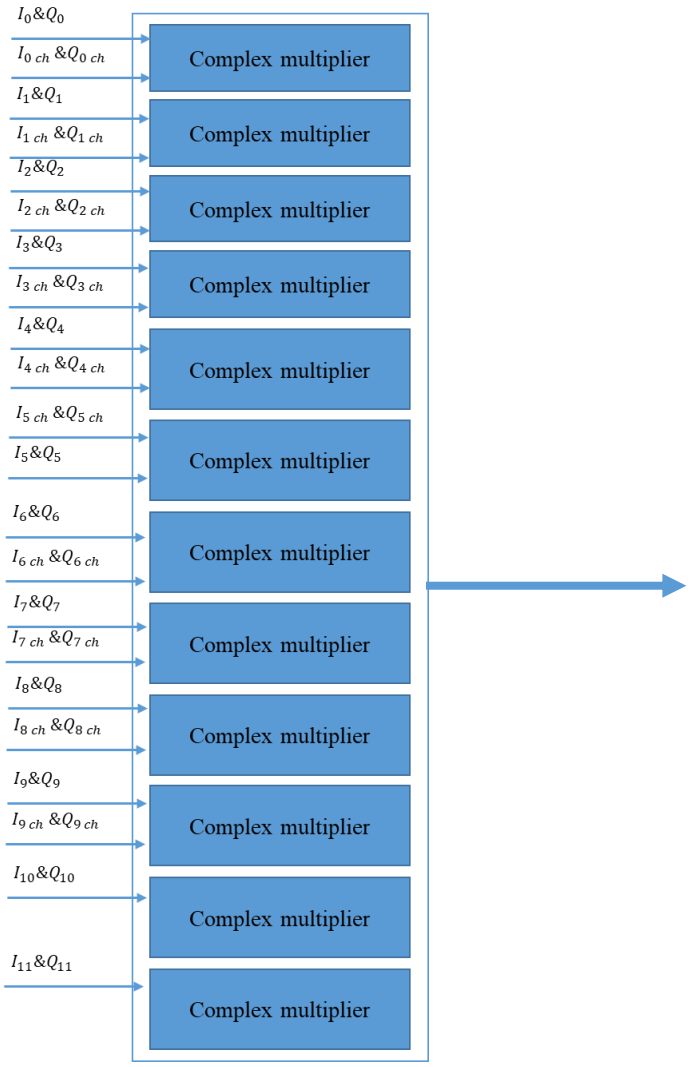
3.8.3. Function of the design:

- Real and imaginary parts of symbol are taken from resource de-mapper block.
- Real and imaginary parts of estimated channel taken from estimation block.
- To get the equalized real and imaginary output, we divide real and imaginary parts of symbol by real and imaginary parts of estimated channel.

3.8.4. Block specification:

- The clock used for the block has period equal 520 ns.
- The block is combinational block.

3.8.5. Detailed block diagram

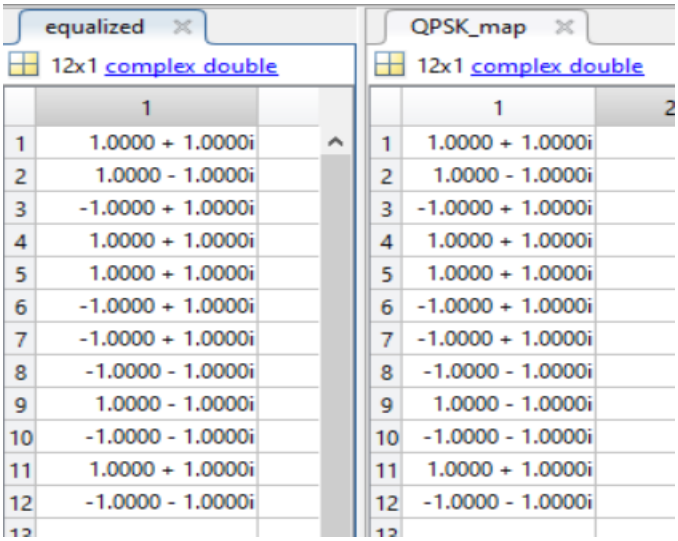


3.8.6. Design Interface:

- 12 value of channel estimation each of them has a real and an imaginary part consists of 16-bits and 1-bit enable from channel estimator.
- 12 value of symbols in frequency domain each of them has a real and an imaginary part consists of 16-bits from resource de-mapper.
- 12 value of equalized symbol each of them has a real and an imaginary part consists of 16-bits and 1-bit done and 4-bit n_slot to P/S and NRS removal block.

3.8.7. Simulation Results:

3.8.7.1. MATLAB Results:



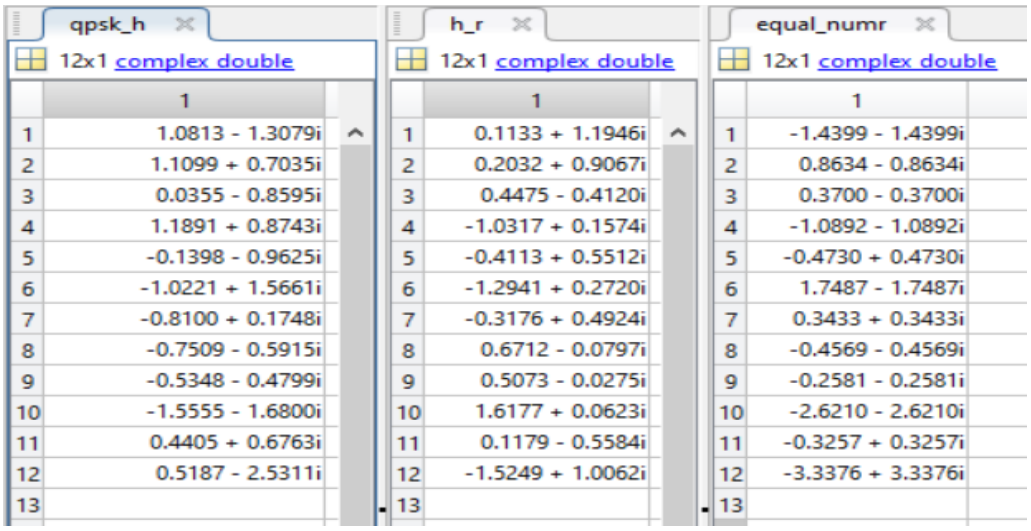
equalized — 12x1 complex double

	1
1	1.0000 + 1.0000i
2	1.0000 - 1.0000i
3	-1.0000 + 1.0000i
4	1.0000 + 1.0000i
5	1.0000 + 1.0000i
6	-1.0000 + 1.0000i
7	-1.0000 + 1.0000i
8	-1.0000 - 1.0000i
9	1.0000 - 1.0000i
10	-1.0000 - 1.0000i
11	1.0000 + 1.0000i
12	-1.0000 - 1.0000i

QPSK_map — 12x1 complex double

	1	2
1	1.0000 + 1.0000i	
2	1.0000 - 1.0000i	
3	-1.0000 + 1.0000i	
4	1.0000 + 1.0000i	
5	1.0000 + 1.0000i	
6	-1.0000 + 1.0000i	
7	-1.0000 + 1.0000i	
8	-1.0000 - 1.0000i	
9	1.0000 - 1.0000i	
10	-1.0000 - 1.0000i	
11	1.0000 + 1.0000i	
12	-1.0000 - 1.0000i	

Figure 142: comparison between input symbols and the output after equalization



qpsk_h — 12x1 complex double

	1
1	1.0813 - 1.3079i
2	1.1099 + 0.7035i
3	0.0355 - 0.8595i
4	1.1891 + 0.8743i
5	-0.1398 - 0.9625i
6	-1.0221 + 1.5661i
7	-0.8100 + 0.1748i
8	-0.7509 - 0.5915i
9	-0.5348 - 0.4799i
10	-1.5555 - 1.6800i
11	0.4405 + 0.6763i
12	0.5187 - 2.5311i
13	

h_r — 12x1 complex double

	1
1	0.1133 + 1.1946i
2	0.2032 + 0.9067i
3	0.4475 - 0.4120i
4	-1.0317 + 0.1574i
5	-0.4113 + 0.5512i
6	-1.2941 + 0.2720i
7	-0.3176 + 0.4924i
8	0.6712 - 0.0797i
9	0.5073 - 0.0275i
10	1.6177 + 0.0623i
11	0.1179 - 0.5584i
12	-1.5249 + 1.0062i
13	

equal_numr — 12x1 complex double

	1
1	-1.4399 - 1.4399i
2	0.8634 - 0.8634i
3	0.3700 - 0.3700i
4	-1.0892 - 1.0892i
5	-0.4730 + 0.4730i
6	1.7487 - 1.7487i
7	0.3433 + 0.3433i
8	-0.4569 - 0.4569i
9	-0.2581 - 0.2581i
10	-2.6210 - 2.6210i
11	-0.3257 + 0.3257i
12	-3.3376 + 3.3376i
13	

Figure 143: inputs and outputs of equalizer

3.8.7.2. RTL Results:

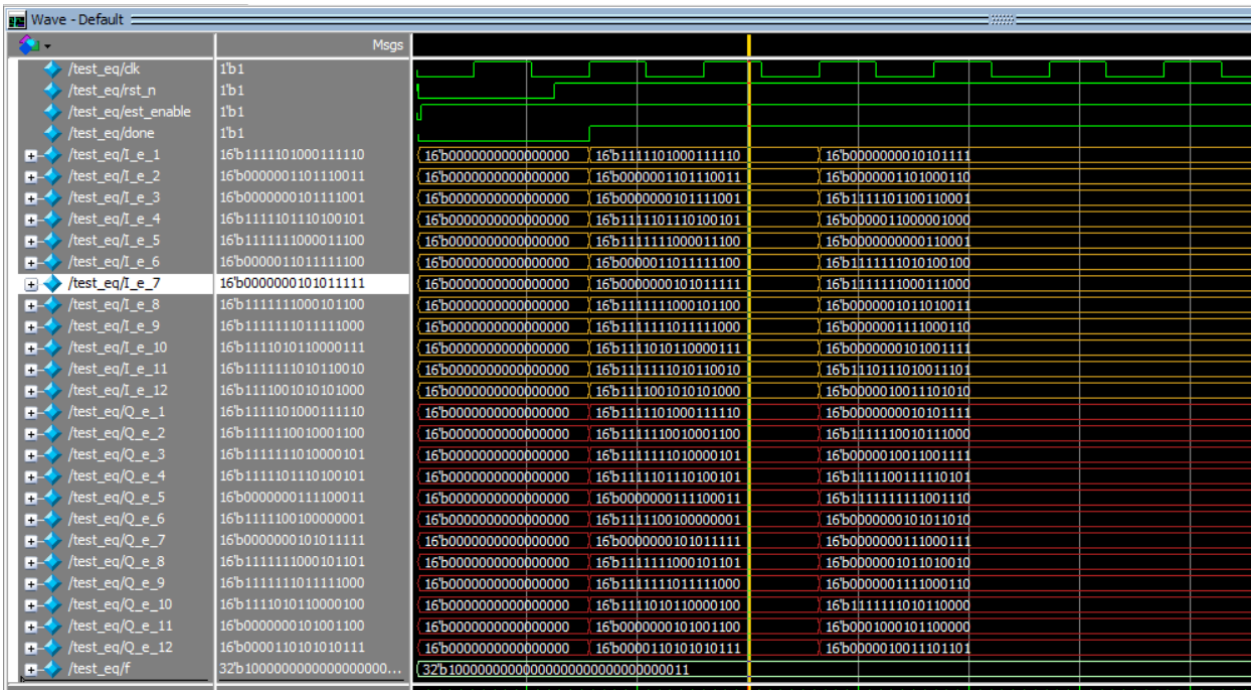


Figure 144: output of RTL of channel Equalizer

In this simulation case the inputs of RTL are the same as the inputs shown in Figure 144 and in the following section we will calculate the average error between MATLAB and RTL simulation.

3.8.7.3. Comparison between MATLAB and RTL

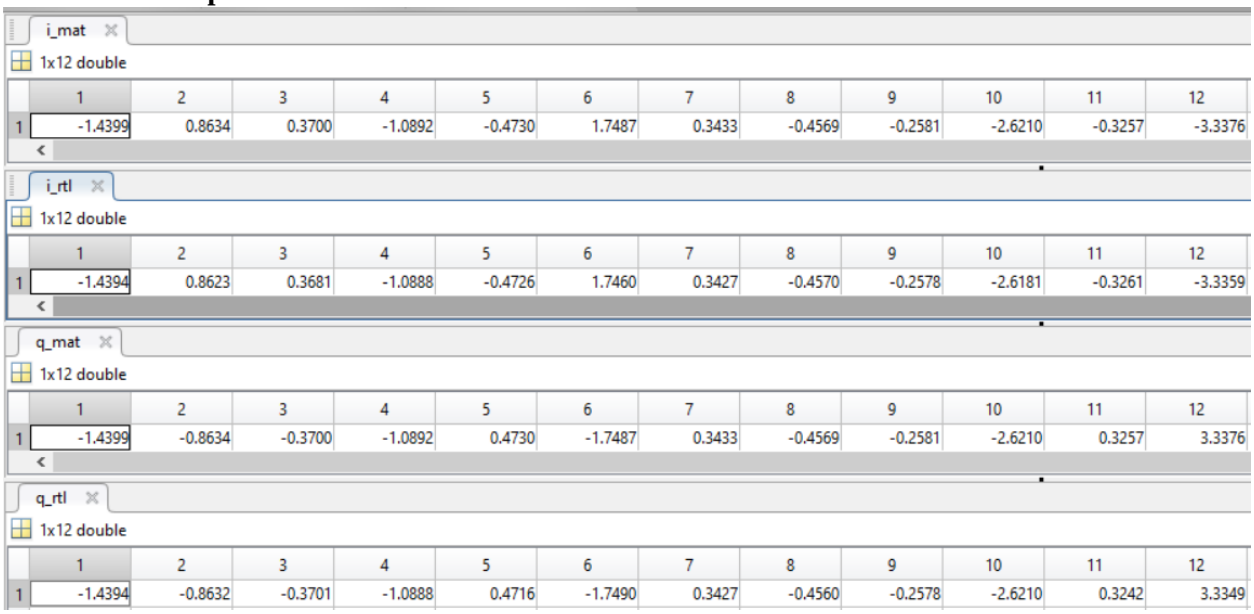


Figure 145: comparison between output from MATLAB and RTL

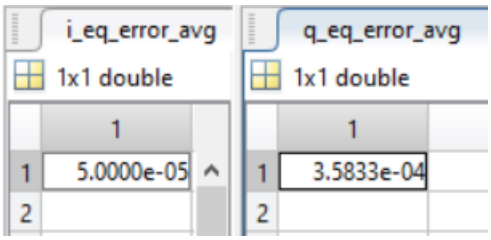


Figure 146: average error between MATLAB and RTL results

3.8.7.4. Synthesis Reports:

```
Number of ports:                         1166
Number of nets:                          1573
Number of cells:                          285
Number of combinational cells:            199
Number of sequential cells:                72
Number of macros:                           0
Number of buf/inv:                         58
Number of references:                      34

Combinational area:       63455.896121
Buf/Inv area:             1395.968013
Noncombinational area:     387.030012
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:          63842.926133
Total area:                 undefined
1
```

Figure 147: area report of channel equalizer

```
  clock clk (rise edge)                                  520.83     520.83
  clock network delay (ideal)                              0.00     520.83
  clock uncertainty                                       -0.50     520.33
  output external delay                                 -257.15     263.19
  data required time                                                263.19
  ---------------------------------------------------------------------------
  data required time                                                263.19
  data arrival time                                                -260.02
  ---------------------------------------------------------------------------
  slack (MET)                                                         3.16


1
```

Figure 148: timing report of channel equalizer

```
Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  44.7325 uW   (56%)
  Net Switching Power  =  35.2868 uW   (44%)
                         ---------
Total Dynamic Power    =  80.0193 uW  (100%)

Cell Leakage Power     = 259.0588 uW


                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power   (   %    )  ,
ttrs
 ------------------------------------------------------------------------------------------
----
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           0.7337        2.2399e-03        1.2831e+03           2.0191  (   0.60%)
sequential         0.0000           0.0000            0.0000            0.0000  (   0.00%)
combinational     43.9993          35.2844        2.5781e+05          337.0931  (  99.40%)
 ------------------------------------------------------------------------------------------
----
Total             44.7330 uW       35.2867 uW     2.5909e+05 nW       339.1122 uW
1
```

Figure 149: power report of channel equalizer

3.8.7.5. Different designs for the equalizer block

There are two different implementation for channel equalizer block the first design is that every clock the equalizer has a new output until all resource de-mapper symbols are equalized and based on it the next block had to operate with clock 12 times of the channel equalizer block in order to out all the parallel data before the new data comes but after integration we found the we have enough time before the resource de-mapper filled with new data. So, we changed the design to make the equalizer get new data to be equalized from resource de-mapper each 12 clock to make the next block operate with same clock. It helps to reduce the output rate of the whole chain as it is a specification for our project.

3.9. Fine Synchronization:

3.9.1. Block Diagram:

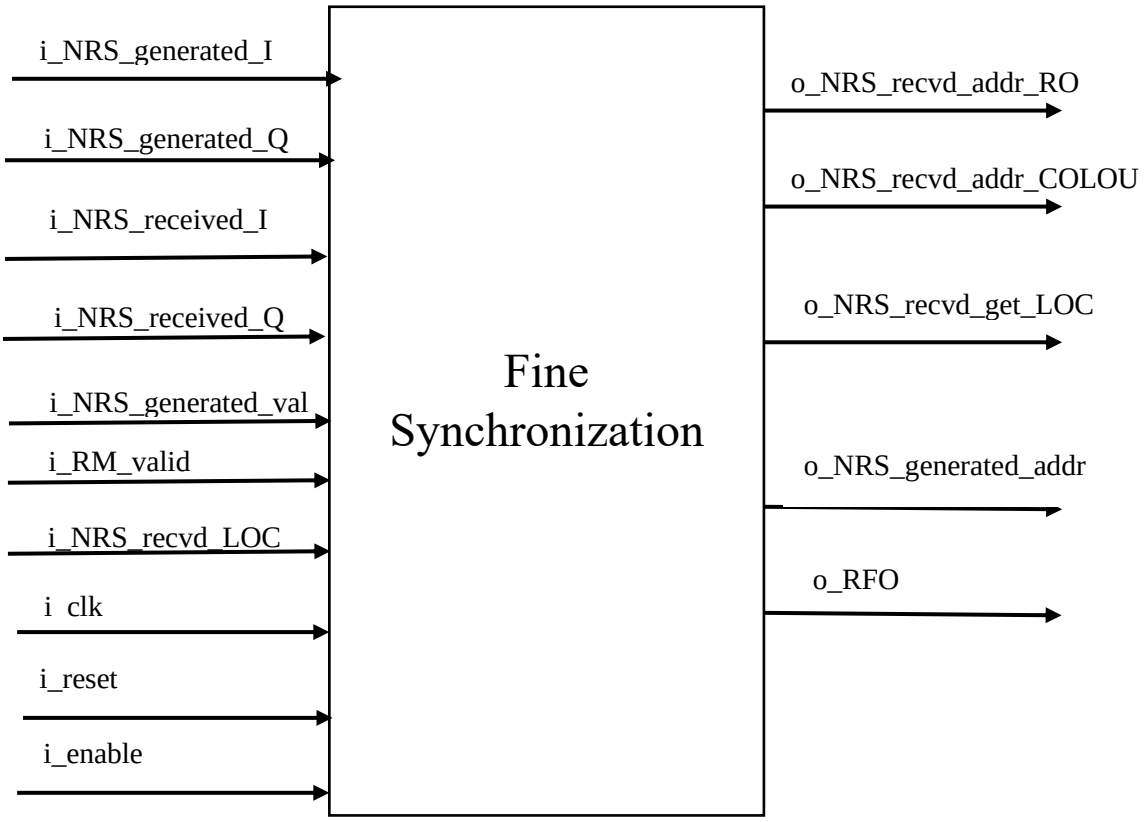


Figure 150 Fine Synchronization Block Diagram

3.9.2. Interface Table:

Table 25 Fine Synch interface table

Signal Name	Direction	Width	Description
$i\_NRS\_generated\_I$	Input	16 bits	NRSs signals comes from NRS generation block
$i\_NRS\_generated\_Q$	Input	16 bits	NRSs signals comes from NRS generation block
$i\_NRS\_recieved\_I$	Input	16 bits	NRSs signals comes from Resource Demapper block
$i\_NRS\_recieved\_Q$	Input	16 bits	NRSs signals comes from Resource Demapper block
i_NRS_generated_valid	Input	1 bit	Enable signal from NRS generation Block
i_RM_valid	Input	1 bit	Enable signal from Resource Demapper Block
i_NRS_recvd_LOC	Input	4 bits	Location of NRS in resource demappper
o_NRS_recvd_get_LOC	Output	3 bits	Address to NRS location Register File to get location.
o_NRS_generated_addr	Output	3 bits	Address to NRS location Register File to get values.
o_NRS_recvd_add_row	Output	4 bits	Row address to Resource demapper
o_NRS_recvd_add_Col	Output	4 bits	Column address to Resource demapper
i_{enable}	Input	1 bit	Enable signal for this block
i_{clk}	Input	1 bit	Positive edge clock
i_{reset}	Input	1 bit	Reset signal for the block equalizer
o_{RFO}	Output	22 bits	Residual Frequency Offset output

3.9.3. Function of the design:

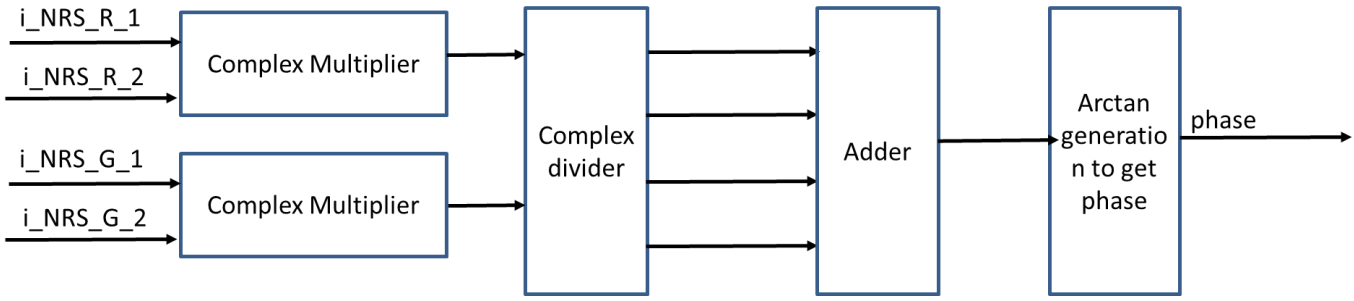


Figure 151 Fine Synchronization Detailed Block Diagram

The main idea of the algorithm is as follow:

1. Conjugate product of the two received NRS signals from resource mapper, and do it again in the two comes from NRS generation.
2. Divide resource mappers product by generation product, this division can be done as complex multiplier by multiply the conjugate of the denominator.
3. Do this 4 times for each subcarrier that includes couple of pilots. And add them together.
4. Finally calculate the phase using inverse tangent theorem.

3.9.3.1. Arctan generation

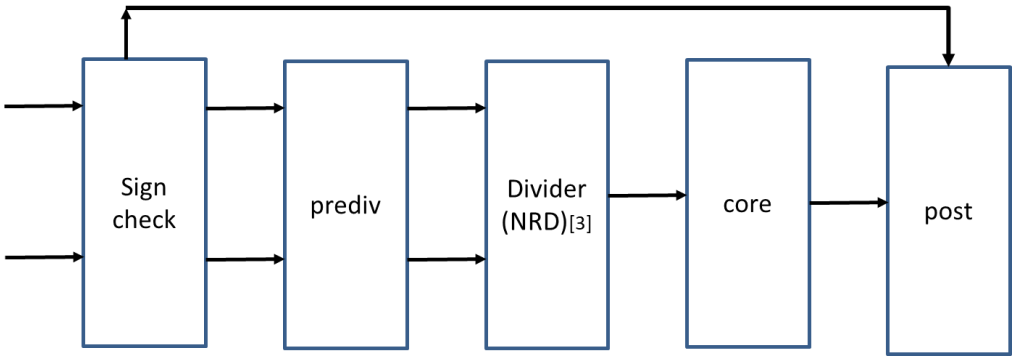


Figure 152 Arctan Block Diagram

We use this algorithm to get $\tan^{-1} z$, the last stage to get both RTO and RFO as shown in equations in 2.9. This architecture aims to avoid high power consumption and long latency [42].

From the nature of arctangent function, it is seems to be liner in the range from $z = [0:1]$ which equivalent to $\theta = [0:45°]$, we will calculate in this range and if the imaginary part is greater than real part we will shift the result by 90°, after that we will map the result angle to one of four quadrant.

We will divide the range, $z = [0:1]$, from into four linear regions to make sure that every region has a constant slope. The segments' slope are chosen to minimize the $Minimum\ Mean\ Square\ Error\ (MMSE)$ between the ideal arctangent function and the approximated one.

$$MMSE = \min\left\{\int_0^1 [\tan^{-1} x - p(z)]^2 dx\right\} \quad (79)$$

Where: $\tan^{-1} x \rightarrow ideal\ arctngent\ function$, $p(z) \rightarrow approximate\ arctangent\ function$

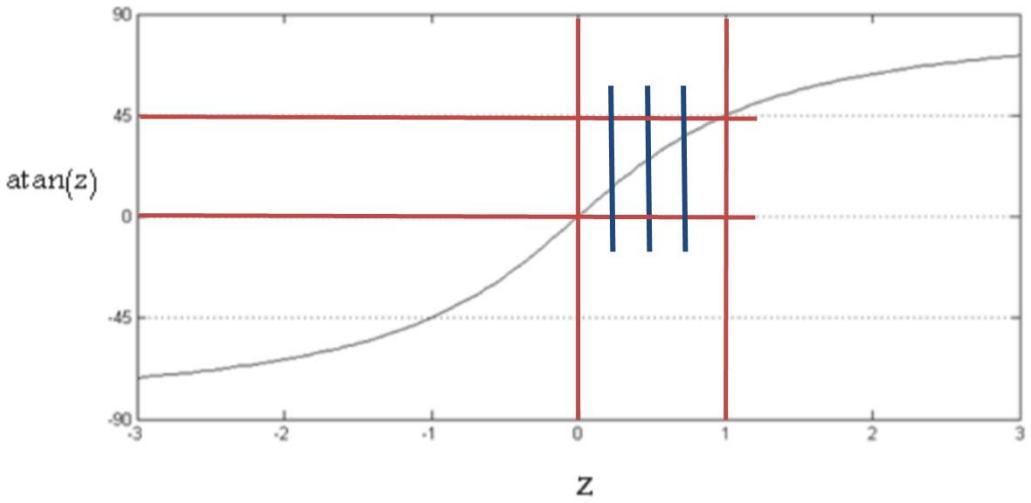


Figure 153 Arctan Curve

3.9.3.1.1. Steps to calculate arctan

Assume that $a = x + jy$

1. We use comparator between x and y, this comparison is used to make sure that the angle will be between $0\ and\ 45°$.
2. We use a divider to calculate z

$$z = \begin{cases} \frac{y}{x}, & x > y \\ \frac{x}{y}, & y > x \end{cases} \tag{80}$$

3. Then substitute in the following equations:

$$\tan^{-1} z = \begin{cases} 56z & , \ 0 < z < 0.25 \\ 50z + 1.5 & , \ 0.25 < z < 0.5 \\ 40z + 6.5 & , \ 0.5 < z < 0.75 \\ 32z + 16 & , \ 0.75 < z < 1 \end{cases} \tag{81}$$

4. If $y > x$:

$$\theta = 90 - \tan^{-1} z \tag{82}$$

3.9.3.1.2. Detailed Block Diagram:

- Sign check Block:
  After receiving real and imaginary parts, this block checks the sign of them and send the absolute values to the next block.
- Prediv Block:
  By receiving the absolute values, this block start comparing both with each other, then it maps the numerator and denominator based on the following:
  if $x1 \geq y1 \rightarrow num. = y1, den = x1$ and so on.
- Divider:
  we used it to divide the two arguments.

- Core:
  the role of this block is to implement the arctan equation, and our implementation does not use any multiplier, so if we need to implement e.g. "$56z$" we will use shifting to left, which is multiply by two i.e. $56z = (z \ll 5) + (z \ll 4) + (z \ll 3)$ and so on.
- Post Block
  as described before, we work in the region$[0: 45°]$, so we need to map the angle if it is larger than$[45°]$. We must steps of mapping:
  1. If $x > y$ the angle will equal the result from equation, and if $x < y$ the angle will equal
     $$90° - \tan^{-1} z$$
  2. Second step is to map to other quadrant based on the signs of numerator and denominator
     $$\theta = \begin{cases} \tan^{-1} z & ,num = +ve\ and\ den = +ve \\ 90 + \tan^{-1} z & ,num = -ve\ and\ den = +ve \\ 180 + \tan^{-1} z & ,num = -ve\ and\ den = -ve \\ 270 + \tan^{-1} z & ,num = +ve\ and\ den = -ve \end{cases} \quad (83)$$

3.9.4. Block Specification:

- The clock used for the block has period equal 130 ns.
- Block latency is 60 clock cycle

3.9.5. Design Interface:

- It takes the received pilots (NRS) and the enable signal from resource mapper.
- Also, in the same time it takes the generated pilots and enable signal from NRS generation block.
- The AND operation between the two enable signals starts the Fine block.
- After calculation, the RFO signal sends to CFO block.

3.9.6. Simulation Results:

3.9.6.1. MATLAB Results:

Variables - NRS_Rec

RFO | atan1 | NRS_Rec

12x14 complex double

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1.0000 + 1.0000i	0.9997 + 1.0003i	0.9993 + 1.0007i	0.9990 + 1.0010i	0.9986 + 1.0014i	0.7058 + 0.7082i	0.9979 + 1.0021i	0.9976 + 1.0024i	0.9972 + 1.0027i	0.9969 + 1.0031i	0.9966 + 1.0034i	0.9962 + 1.0038i	0.7099 - 0.7041i	0.9955 + 1.0045i
2	1.0000 - 1.0000i	1.0003 - 0.9997i	1.0007 - 0.9993i	1.0010 - 0.9990i	1.0014 - 0.9986i	1.0017 - 0.9983i	1.0021 - 0.9979i	1.0024 - 0.9976i	1.0027 - 0.9972i	1.0031 - 0.9969i	1.0034 - 0.9966i	1.0038 - 0.9962i	1.0041 - 0.9959i	1.0045 - 0.9955i
3	-1.0000 + 1.0000i	-1.0003 + 0.9997i	-1.0007 + 0.9993i	-1.0010 + 0.9990i	-1.0014 + 0.9986i	-1.0017 + 0.9983i	-1.0021 + 0.9979i	-1.0024 + 0.9976i	-1.0027 + 0.9972i	-1.0031 + 0.9969i	-1.0034 + 0.9966i	-1.0038 + 0.9962i	-1.0041 + 0.9959i	-1.0045 + 0.9955i
4	1.0000 - 1.0000i	1.0003 - 0.9997i	1.0007 - 0.9993i	1.0010 - 0.9990i	1.0014 - 0.9986i	1.0017 - 0.9983i	-0.7055 - 0.7085i	1.0024 - 0.9976i	1.0027 - 0.9972i	1.0031 - 0.9969i	1.0034 - 0.9966i	1.0038 - 0.9962i	1.0041 - 0.9959i	0.7102 - 0.7038i
5	1.0000 - 1.0000i	1.0003 - 0.9997i	1.0007 - 0.9993i	1.0010 - 0.9990i	1.0014 - 0.9986i	1.0017 - 0.9983i	1.0021 - 0.9979i	1.0024 - 0.9976i	1.0027 - 0.9972i	1.0031 - 0.9969i	1.0034 - 0.9966i	1.0038 - 0.9962i	1.0041 - 0.9959i	1.0045 - 0.9955i
6	-1.0000 + 1.0000i	-1.0003 + 0.9997i	-1.0007 + 0.9993i	-1.0010 + 0.9990i	-1.0014 + 0.9986i	-1.0017 + 0.9983i	-1.0021 + 0.9979i	-1.0024 + 0.9976i	-1.0027 + 0.9972i	-1.0031 + 0.9969i	-1.0034 + 0.9966i	-1.0038 + 0.9962i	-1.0041 + 0.9959i	-1.0045 + 0.9955i
7	-1.0000 + 1.0000i	-1.0003 + 0.9997i	-1.0007 + 0.9993i	-1.0010 + 0.9990i	-1.0014 + 0.9986i	-0.7082 + 0.7058i	-1.0021 + 0.9979i	-1.0024 + 0.9976i	-1.0027 + 0.9972i	-1.0031 + 0.9969i	-1.0034 + 0.9966i	-1.0038 + 0.9962i	-0.7041 - 0.7099i	-1.0045 + 0.9955i
8	1.0000 + 1.0000i	0.9997 + 1.0003i	0.9993 + 1.0007i	0.9990 + 1.0010i	0.9986 + 1.0014i	0.9983 + 1.0017i	0.9979 + 1.0021i	0.9976 + 1.0024i	0.9972 + 1.0027i	0.9969 + 1.0031i	0.9966 + 1.0034i	0.9962 + 1.0038i	0.9959 + 1.0041i	0.9955 + 1.0045i
9	1.0000 + 1.0000i	0.9997 + 1.0003i	0.9993 + 1.0007i	0.9990 + 1.0010i	0.9986 + 1.0014i	0.9983 + 1.0017i	0.9979 + 1.0021i	0.9976 + 1.0024i	0.9972 + 1.0027i	0.9969 + 1.0031i	0.9966 + 1.0034i	0.9962 + 1.0038i	0.9959 + 1.0041i	0.9955 + 1.0045i
10	1.0000 - 1.0000i	1.0003 - 0.9997i	1.0007 - 0.9993i	1.0010 - 0.9990i	1.0014 - 0.9986i	1.0017 - 0.9983i	0.7055 + 0.7085i	1.0024 - 0.9976i	1.0027 - 0.9972i	1.0031 - 0.9969i	1.0034 - 0.9966i	1.0038 - 0.9962i	1.0041 - 0.9959i	0.7038 + 0.7102i
11	-1.0000 + 1.0000i	-1.0003 + 0.9997i	-1.0007 + 0.9993i	-1.0010 + 0.9990i	-1.0014 + 0.9986i	-1.0017 + 0.9983i	-1.0021 + 0.9979i	-1.0024 + 0.9976i	-1.0027 + 0.9972i	-1.0031 + 0.9969i	-1.0034 + 0.9966i	-1.0038 + 0.9962i	-1.0041 + 0.9959i	-1.0045 + 0.9955i
12	1.0000 + 1.0000i	0.9997 + 1.0003i	0.9993 + 1.0007i	0.9990 + 1.0010i	0.9986 + 1.0014i	0.9983 + 1.0017i	0.9979 + 1.0021i	0.9976 + 1.0024i	0.9972 + 1.0027i	0.9969 + 1.0031i	0.9966 + 1.0034i	0.9962 + 1.0038i	0.9959 + 1.0041i	0.9955 + 1.0045i
13														

Figure 154 NRS in subframe

1	2	3	4
0.7071 + 0.7071i	-0.7071 - 0.7071i	0.7071 - 0.7071i	0.7071 - 0.7071i
-0.7071 + 0.7071i	0.7071 + 0.7071i	-0.7071 - 0.7071i	0.7071 + 0.7071i

Figure 155 Values of NRS

```
Command Window
>> frequency_offset

frequency_offset =

   105

>> z

z =

   3.9987 - 0.0096i

>> atanOut

atanOut =

    0.1347

>> Offset_o

Offset_o =

  103.4466
```

Figure 156 Fine Synchronization MATLAB Output

3.9.6.2. RTL Results:

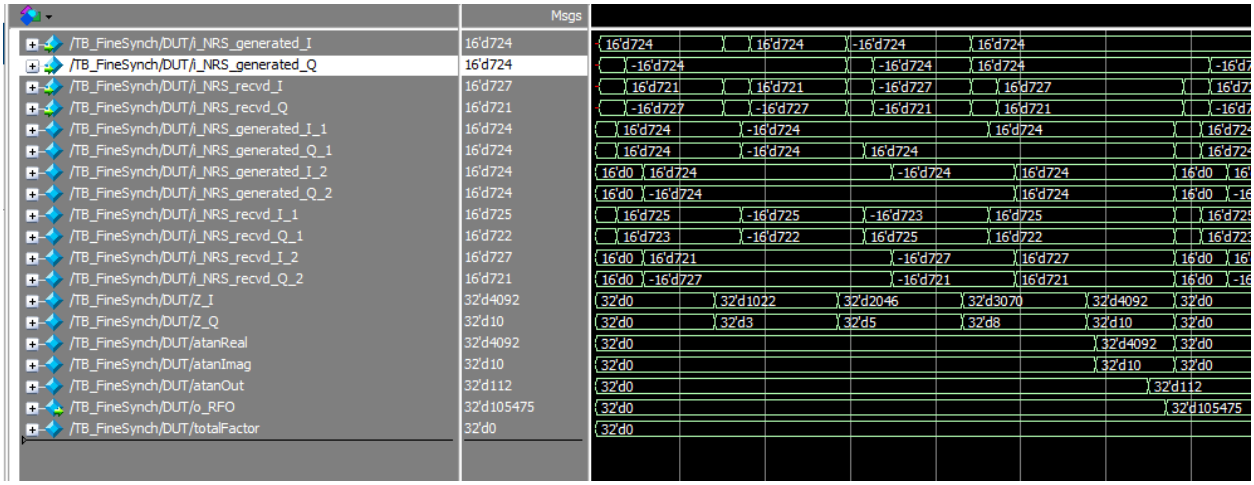


Figure 157 Fine Synchronization RTL Output

3.9.6.3. Synthesis Reports:

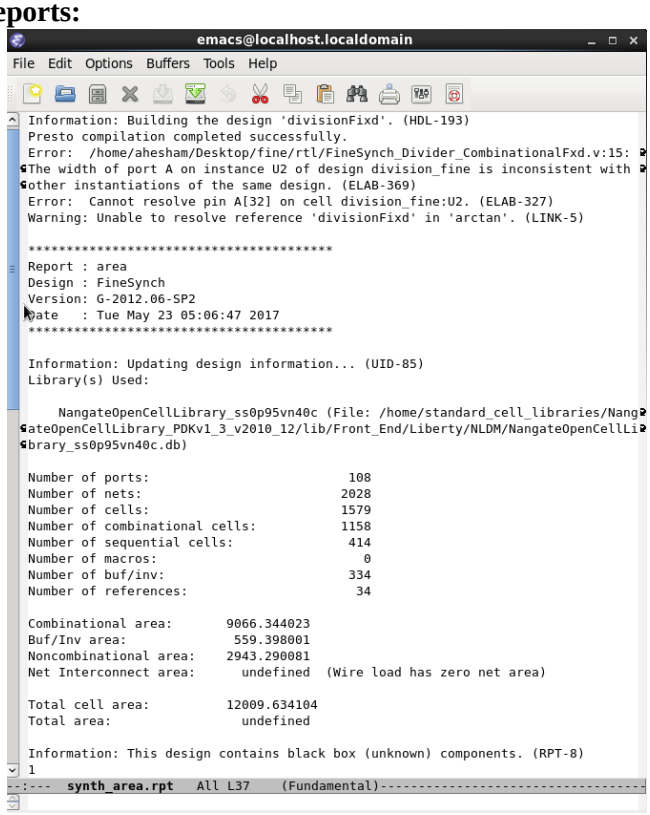


Figure 158 Fine Synchronization Area Report

```
Operating Conditions: worst_low   Library: NangateOpenCellLibrary_ss0p95vn40c
Wire Load Model Mode: top

Design        Wire Load Model            Library
------------------------------------------------
FineSynch              5K_hvratio_1_1    NangateOpenCellLibrary_ss0p95vn40c


Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  11.3569 uW   (94%)
  Net Switching Power  = 762.8978 nW    (6%)
                         ---------
Total Dynamic Power    =  12.1198 uW  (100%)

Cell Leakage Power     =  53.1423 uW


                 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          11.0535        3.4199e-02        9.9242e+03           21.0119  (  32.20%)
sequential         0.0000           0.0000            0.0000            0.0000  (   0.00%)
combinational      0.3034           0.7287        4.3218e+04           44.2503  (  67.80%)
------------------------------------------------------------------------------------------------
Total             11.3569 uW         0.7629 uW     5.3142e+04 nW        65.2622 uW
1
```

Figure 159 Fine Synchronization Power Report

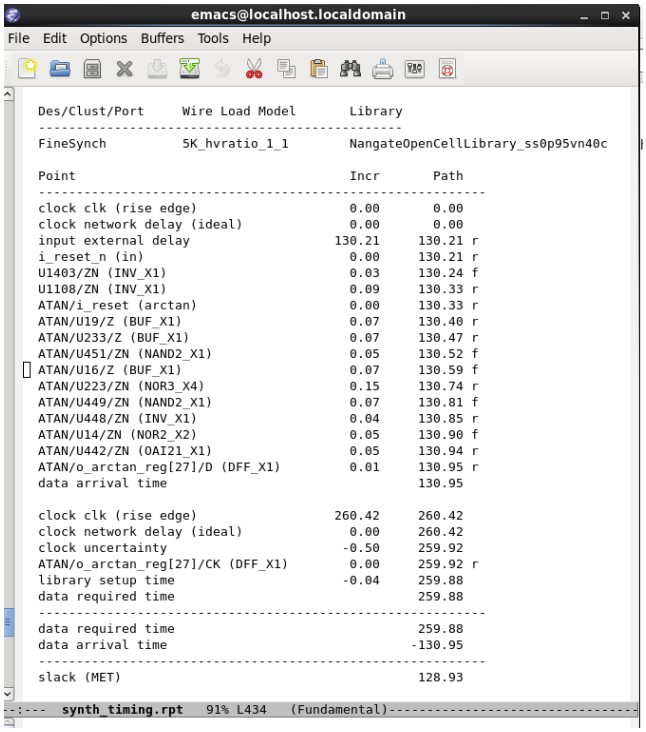
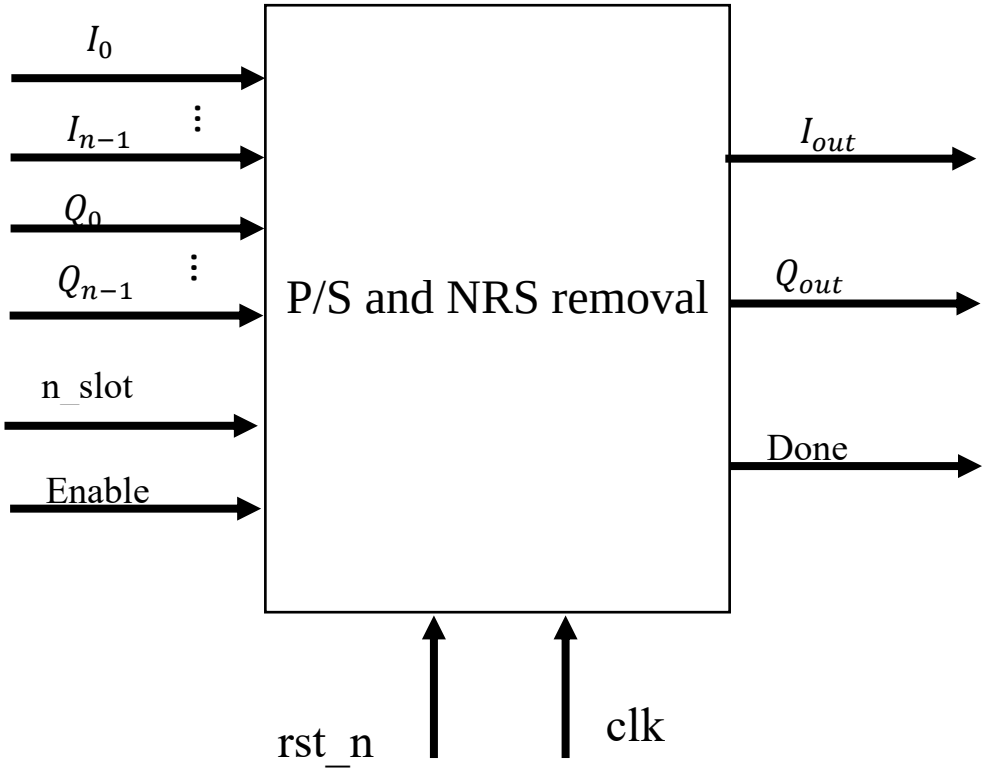


Figure 160 Fine Synchronization Timing Report

3.10. P/S and NRS removal:

3.10.1. Block Diagram:



3.10.2. Interface Table:

Table 26 P/S and NRS removal interface table

Signal Name	Direction	Width	Description
$\mathbf{I_n}$	Input	16	Symbol real part
$\mathbf{Q_n}$	Input	16	Symbol imaginary part
Enable	Input	1	P/S and NRS removal enable
clk	Input	1	P/S and NRS removal clock
Rst_n	Input	1	P/S and NRS removal reset
$\mathbf{I_{out}}$	Output	16	Symbol real part
$\boldsymbol{Q_{out}}$	Output	16	Symbol imaginary part
Done	Output	1	Signal indicate that P/S and NRS removal is done
n_slot	Input	4	Slot number
nrs_LOC_done	Input	1	Enable from channel estimation
Pilot1	Input	4	Location of the first pilot
Pilot2	Input	4	Location of the second pilot
Pilot3	Input	4	Location of the third pilot
Pilot4	Input	4	Location of the fourth pilot

3.10.3. Function of the design:

The function of the design that it changes the parallel input in serial outputs and removing the NRS symbols.

3.10.4. Design specification:

- The clock that the block used has period equal 520 ns.
- Latency of the block is 1 clock cycle.

3.10.5. Design Interface:

- 12 value of equalized symbol each of them has a real and an imaginary part consists of 16-bits and 1-bit enable and 4-bit n_slot from channel equalizer block.
- Real and imaginary part of the symbol each consists of 16-bit and 1-bit done signal to De-modulation block.
- 1-bit nrs_LOC_done and 4-bit pilot1, pilot2, pilot3 and pilot4 from channel estimation block.

3.10.6. Simulation Results:

3.10.6.1. RTL Results:

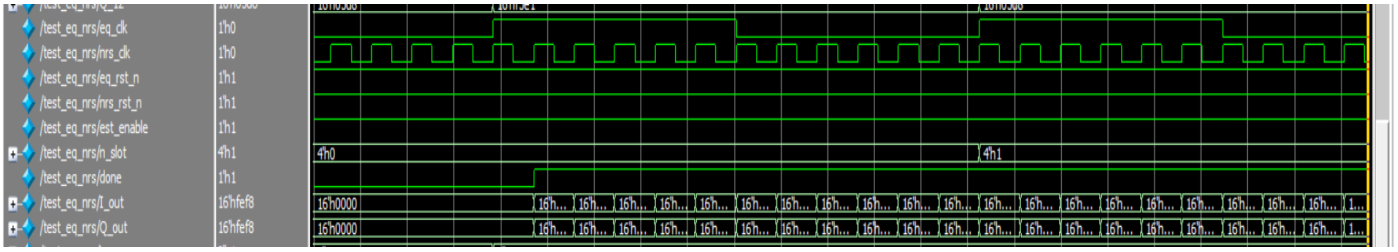


Figure 161: output of P/S and NRS removal block

The previous figure shows the output of the block at slot number equal 1 that doesn't contain NRS symbol.

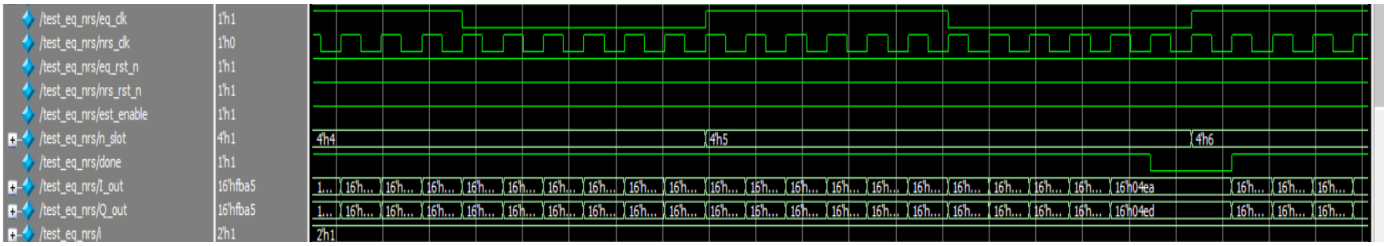


Figure 162: output of P/S and NRS removal

The previous figure shows the output of the block at slot number equal 5 which has 2 NRS symbols so the output will be 10 serial symbols only as shown.

3.10.6.2. Synthesis Reports:

```
    NangateOpenCellLibrary_ss0p95vn40c (File: /home/standard_cell_libraries/Nang
ateOpenCellLibrary_PDKv1_3_v2010_12/lib/Front_End/Liberty/NLDM/NangateOpenCellLi
brary_ss0p95vn40c.db)

Number of ports:                          441
Number of nets:                          2881
Number of cells:                         2077
Number of combinational cells:           1647
Number of sequential cells:               430
Number of macros:                           0
Number of buf/inv:                        280
Number of references:                      21

Combinational area:       1975.847988
Buf/Inv area:              186.199998
Noncombinational area:    1974.783942
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:          3950.631930
Total area:                 undefined
1
```

Figure 163: area report of P/S and NRS removal block

```
Des/Clust/Port     Wire Load Model       Library
------------------------------------------------
ps                 5K_hvratio_1_1        NangateOpenCellLibrary_ss0p95vn40c

Point                                    Incr       Path
-----------------------------------------------------------
clock clk (rise edge)                    0.00       0.00
clock network delay (ideal)              0.00       0.00
input external delay                    21.70      21.70 r
eq_enable (in)                           0.00      21.70 r
U3047/ZN (INV_X1)                        0.02      21.72 f
U2245/ZN (NOR2_X1)                       0.05      21.77 r
U2170/ZN (INV_X1)                        0.03      21.80 f
U2076/ZN (OR2_X1)                        0.07      21.87 f
U2077/ZN (INV_X1)                        0.10      21.97 r
U3045/ZN (AOI21_X1)                      0.06      22.03 f
U3043/ZN (OAI21_X1)                      0.04      22.06 r
i_reg[1]/D (DFFR_X1)                     0.01      22.07 r
data arrival time                                  22.07

clock clk (rise edge)                   43.40      43.40
clock network delay (ideal)              0.00      43.40
clock uncertainty                       -0.50      42.90
i_reg[1]/CK (DFFR_X1)                    0.00      42.90 r
library setup time                      -0.04      42.87
data required time                                 42.87
-----------------------------------------------------------
data required time                                 42.87
data arrival time                                 -22.07
-----------------------------------------------------------
slack (MET)                                        20.80
```

Figure 164: timing report of P/S and NRS removal block

```
Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  34.2482 uW   (87%)
  Net Switching Power  =   5.2888 uW   (13%)
                         ---------
Total Dynamic Power    =  39.5370 uW  (100%)

Cell Leakage Power     =  15.9302 uW


                 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          30.0688           0.5330        7.2426e+03           37.8445  (  68.23%)
sequential         0.0000           0.0000            0.0000            0.0000  (   0.00%)
combinational      4.1794           4.7558        8.6876e+03           17.6227  (  31.77%)
--------------------------------------------------------------------------------------------------
Total             34.2482 uW        5.2888 uW     1.5930e+04 nW        55.4672 uW
1
```

Figure 165: power report of P/S and NRS removal block

3.11. De-Modulation:

3.11.1. Block Diagram:

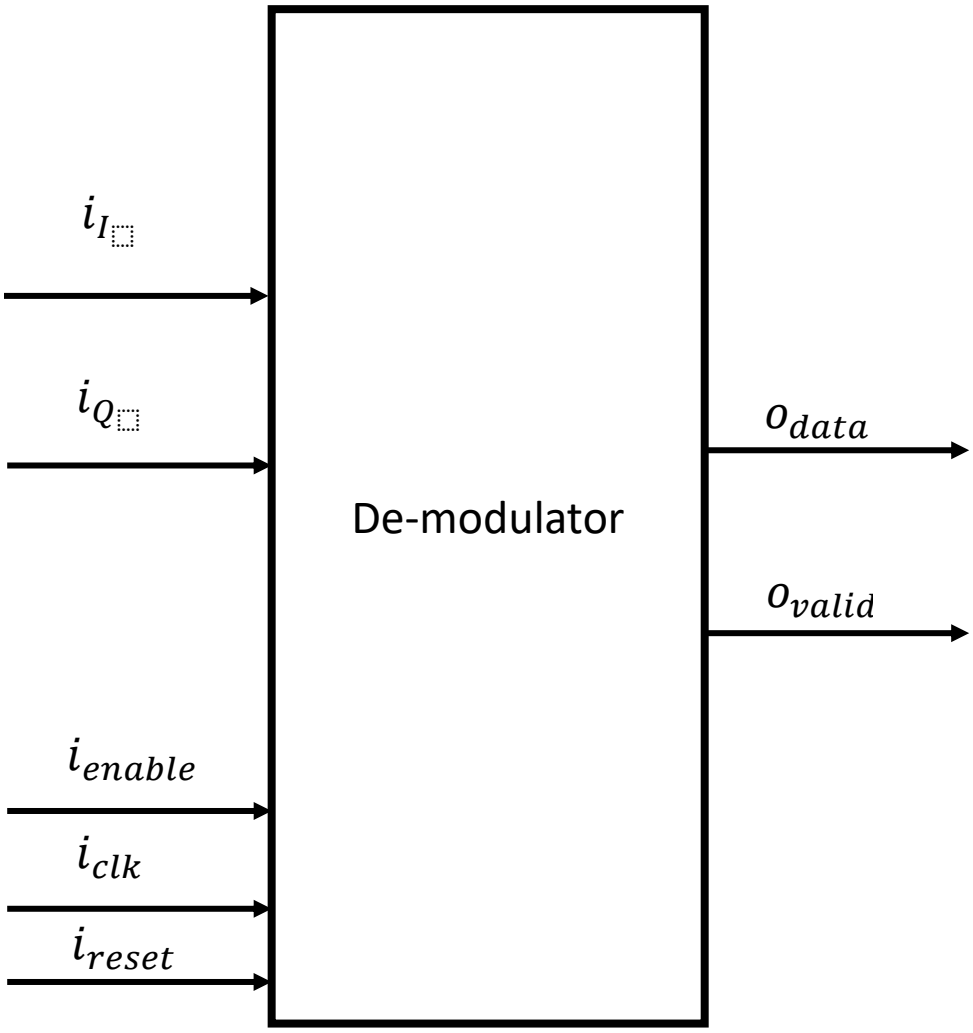


Figure 166 Demapper Block Diagram

3.11.2. Interface Table:

Table 27 De-modulator interface table

Signal Name	Direction	Width	Description
i_I	Input	1 bit	I, serial input comes from P/S & NRS removal block
i_Q	Input	1 bit	Q, serial input comes from P/S & NRS removal block
i_{enable}	Input	1 bit	Enable signal comes from P/S & NRS removal block.
i_{clk}	Input	1 bit	Positive edge clock
i_{resest}	Input	1 bit	Asynchronous reset
o_{data}	Output	1 bit	Serial output de-mapped data
o_{valid}	Output	1 bit	Validation signal for the next block

3.11.3. Function of the design

- The symbol de-mapper block takes the most significant bit (MSB) of each bus I, Q comes from P/S & NRS removal block then the symbol will be de-mapped based on its sign.
- The inputs of this block are two buses each of them sends 12 serial bits.
- The de-mapper output is serial bits, so it works with double rate of P/S & NRS removal block. As shown in figure 148.

3.11.4. Design Interface

- As P/S & NRS removal block starts to send it will rise a done signal and it will enable de-mapper block.
- If the de-mapper is enabled, it will enable de-scrambler block.

3.11.5. Simulation Results:

As shown in figure 146, the inputs to de-modulation block and in figure 147 is the output from de-modulation block. And in figure 148 is the same inputs from MATLAB but in fixed point format and in figure 148 is the results from RTL for the same inputs.

3.11.5.1. MATLAB Results:

2x160 double

	1	2	3	4	5	6	7	8	9	10	11
1	-0.0195	-0.0382	-0.1083	-0.0202	0.1088	-0.1763	0.1072	0.0866	0.0504	0.0917	-0.0663
2	0.0507	-0.0822	0.1189	0.1065	0.0722	0.2214	-0.1789	-0.2068	-0.2283	-0.2581	0.3181
3											

Figure 167 Input to demapper from p/s & NRS removal

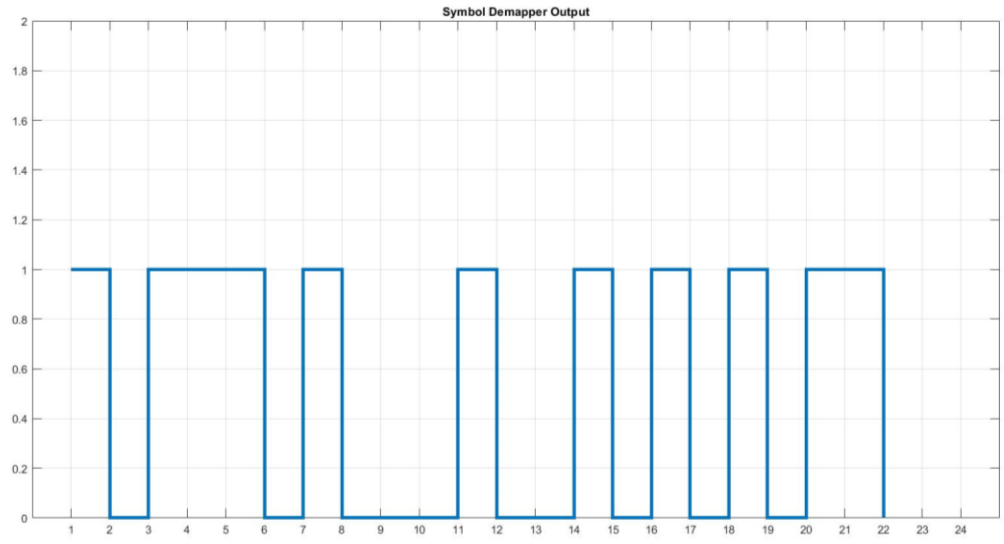


Figure 168 MATLAB Output from Demapper

3.11.5.2. RTL Results

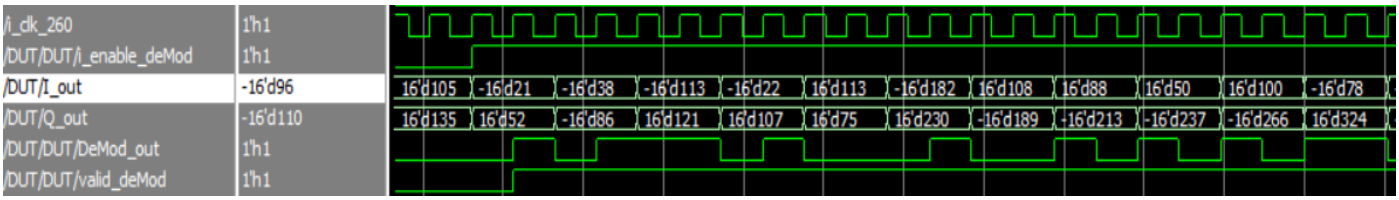


Figure 169 RTL Output from Demapper

3.11.5.3. Synthesis Results

```
****************************************
Report : area
Design : DeModulator
Version: G-2012.06-SP2
Date   : Wed May  3 21:12:08 2017
****************************************

Information: Updating design information... (UID-85)
Library(s) Used:

    NangateOpenCellLibrary_ss0p95vn40c (File: /home/standard_cell_libraries/NangateOpenCellLibrary_PDKv1_3_v2010_12/lib/Front_End/Liberty/NLDM/NangateOpenCellLibrary_ss0p95vn40c.db)

Number of ports:                            7
Number of nets:                            21
Number of cells:                           14
Number of combinational cells:             11
Number of sequential cells:                 3
Number of macros:                           0
Number of buf/inv:                          4
Number of references:                       9

Combinational area:         10.640000
Buf/Inv area:                2.128000
Noncombinational area:      15.162000
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:            25.802000
Total area:                 undefined
1
```

Figure 170 Demapper Area Report

```
Operating Conditions: worst_low   Library: NangateOpenCellLibrary_ss0p95vn40c
Wire Load Model Mode: top

Design        Wire Load Model            Library
------------------------------------------------
DeModulator            5K_hvratio_1_1    NangateOpenCellLibrary_ss0p95vn40c


Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  65.5902 nW   (87%)
  Net Switching Power  =  10.2078 nW   (13%)
                         ---------
Total Dynamic Power    =  75.7980 nW  (100%)

Cell Leakage Power     = 111.1573 nW


                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       5.5901e-02       3.2851e-03           53.6927            0.1129  (  60.38%)
sequential         0.0000           0.0000            0.0000            0.0000  (   0.00%)
combinational  9.6889e-03       6.9227e-03           57.4646        7.4076e-02  (  39.62%)
------------------------------------------------------------------------------------------------
Total          6.5590e-02 uW     1.0208e-02 uW       111.1573 nW         0.1870 uW
1
```

Figure 171 Demapper Power Report

```
  Startpoint: o_data_deMod_reg
              (rising edge-triggered flip-flop clocked by clk)
  Endpoint: o_data_deMod
            (output port clocked by clk)
  Path Group: clk
  Path Type: max

  Des/Clust/Port     Wire Load Model       Library
  ------------------------------------------------
  DeModulator        5K_hvratio_1_1        NangateOpenCellLibrary_ss0p95vn40c

  Point                                    Incr       Path
  -----------------------------------------------------------
  clock clk (rise edge)                    0.00       0.00
  clock network delay (ideal)              0.00       0.00
  o_data_deMod_reg/CK (DFFR_X1)            0.00       0.00 r
  o_data_deMod_reg/Q (DFFR_X1)             0.09       0.09 r
  o_data_deMod (out)                       0.00       0.09 r
  data arrival time                                   0.09

  clock clk (rise edge)                  260.42     260.42
  clock network delay (ideal)              0.00     260.42
  clock uncertainty                       -0.50     259.92
  output external delay                 -130.21     129.71
  data required time                                129.71
  -----------------------------------------------------------
  data required time                                129.71
  data arrival time                                  -0.09
  -----------------------------------------------------------
  slack (MET)                                       129.62


1
```

Figure 172 Demapper Timing Report

3.12. Descrambling:

3.12.1. Block Diagram:

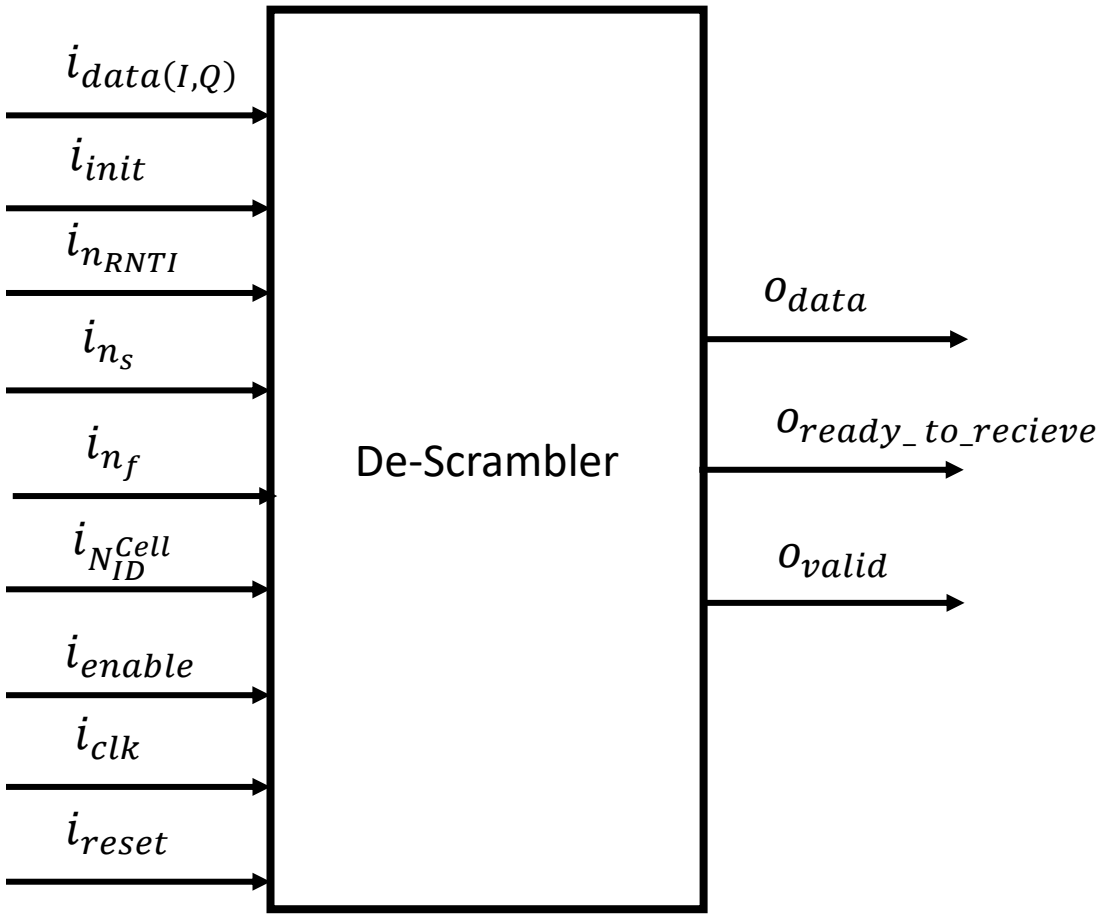


Figure 173 Descrambling Block Diagram

3.12.2. Interface Table:

Table 28 Descrambler Interface table

Signal Name	Direction	Width	Description
$i_{data(I,Q)}$	Input	1 bit	I, Q as a serial input comes from de-mapper
i_{init}	Input	1 bit	Init signal of LFSRs initialization
$i_{n_{RNTI}}$	Input	16 bits	Radio Network Temporary Identifier signal
i_{n_s}	Input	1 bit	Value first slot of transmission
i_{n_f}	Input	1 bit	Value first frame of transmission
$i_{N_{ID}^{Cell}}$	Input	9 bits	Cell identifier
i_{clk}	Input	1 bit	Positive edge clock
i_{enable}	Input	1 bit	Enable signal comes valid out signal from de-mapping
o_{data}	Output	1 bit	Serial output de-scrambled data
$o_{ready\_to\_recieve}$	Output	1 bit	Signal to equalizer block to indicate it to start working
o_{valid}	Output	1 bit	Validation signal for the next block

3.12.3. Function of the design:

- It takes the initial parameters from upper layer then start to initiate the gold sequence, initialization takes almost 1600 clock cycle. To solve the issue in timing due to 1600 clock cycle. We decided to start the initialization of the gold sequence as resource de-mapper finishes.
- The gold sequence gets generated -after initialization- with code word length Mpn.
- The input data get XORed with gold sequence.
- The gold sequence after finishing the Mpn, the gold sequence reinitiates again to its point.

3.12.4. Design Interface:

- The resource de-mapper gives frame and sub-frame numbers then when the resource de-mapper finished and rises its done signal the De-scrambling starts to work.
- After initialization, the descrambling sends a ready signal to channel equalizer to control its work within the time of resource de-mapper existing.
- The interface of de-scrambler with the previous block –de-modulation-, the output of de-modulation is 24 serial bit which is the code word Mpn and the rate of the de-scrambler is equal to the rate of the de-modulation.
- There are two cases of stopping descrambling both of them are related to the enable of the demodulating but the descrambling sense the length of the time slot in the resource de-mapper, if it is 10 symbols, it stops descrambling and save the remains code to the next slot as the NRS signals doesn't descrambled.
- The interface with the next block –rate de-matcher-, once the descrambler starts its operation, it sends a valid signal to rate de-matcher.

3.12.5. Simulation Results:

As shown in figure 153, the inputs to de-scrambling block and in figures 154, 155, 156, 157 are the output from de-scrambling for different cases. And in figures 158, 159, 160, 161 is the same inputs from MATLAB and in figure 158, 159, 160, 161 are the results from RTL for the same inputs.

3.12.5.1. MATLAB Results:

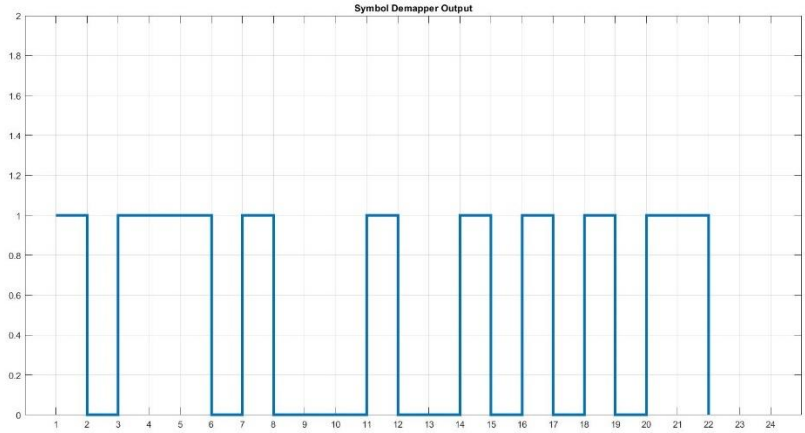


Figure 174 Input to Descrambling from De-mapper

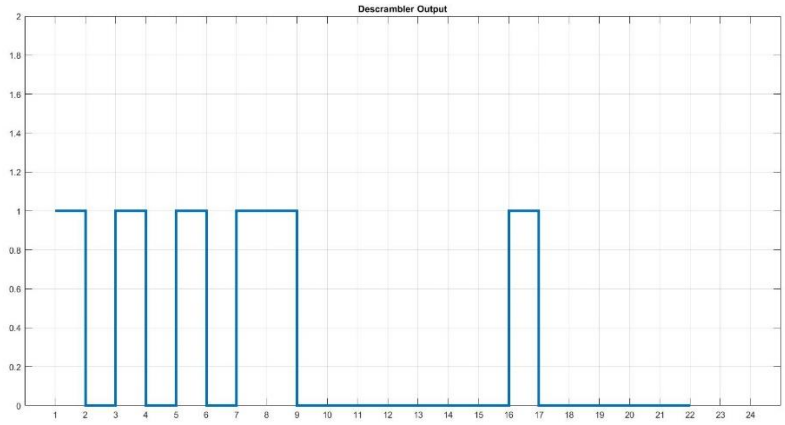


Figure 175 MATLAB Output from Descrambling for NcellID = 0, Ns = 2, Nf = 100

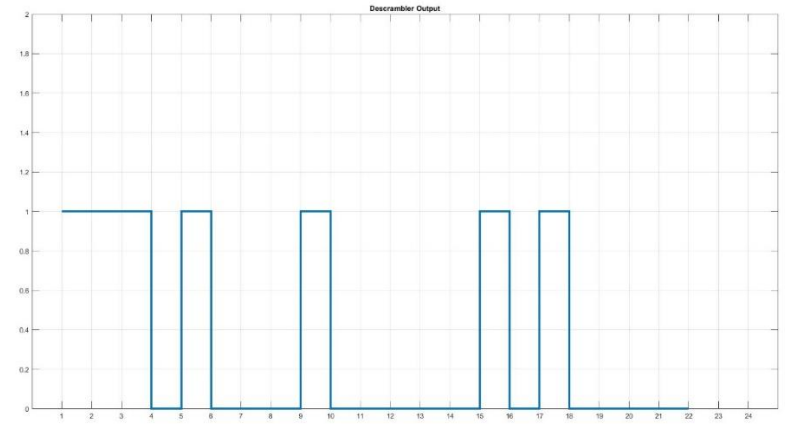


Figure 176 MATLAB Output from Descrambling for NcellID = 0, Ns = 3, Nf = 100

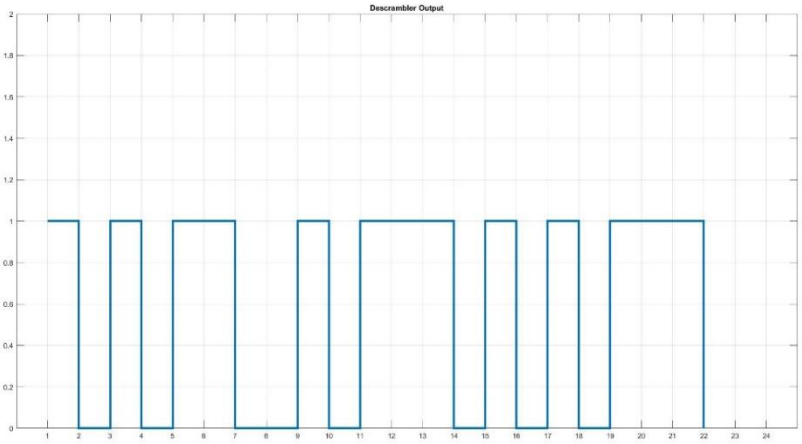


Figure 177 MATLAB Output from Descrambling for NcellID = 0, Ns = 3, Nf = 101

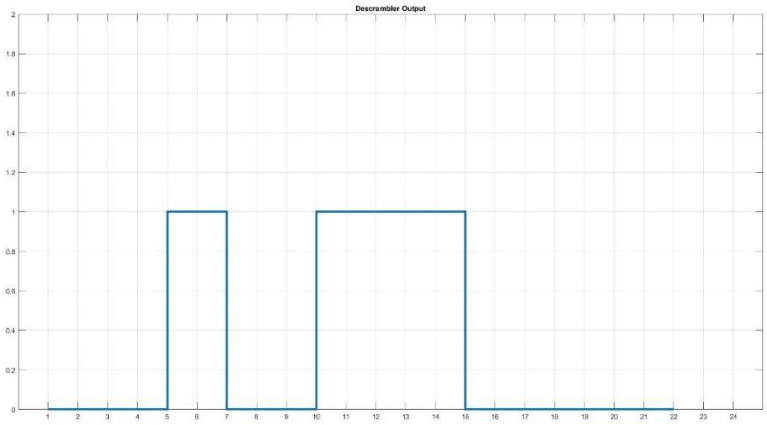


Figure 178 MATLAB Output from Descrambling for NcellID = 14, Ns = 3, Nf = 101

3.12.5.2. RTL Results:

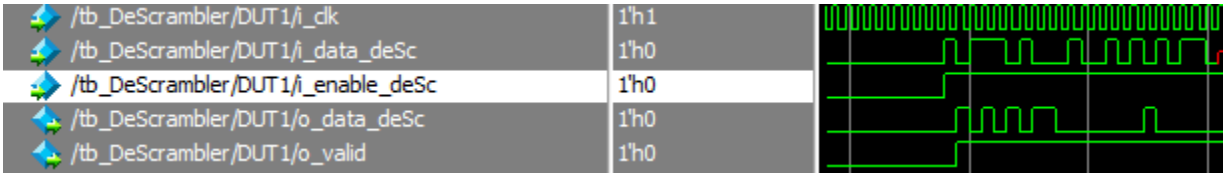


Figure 179 RTL Output from Descrambling for NcellID = 0, Ns = 2, Nf = 100

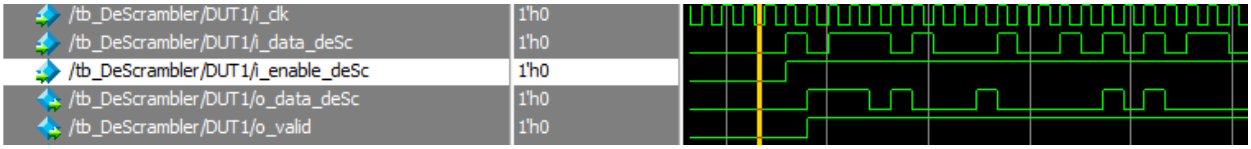


Figure 180 RTL Output from Descrambling for NcellID = 0, Ns = 3, Nf = 100

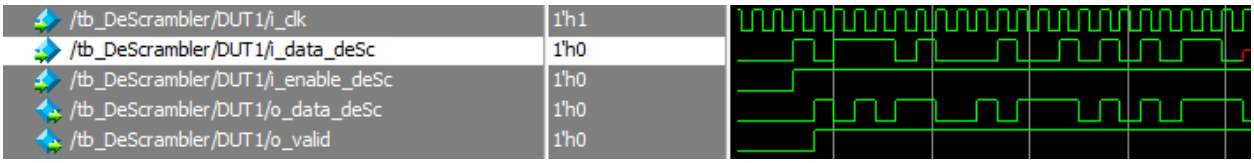


Figure 181 RTL Output from Descrambling for NcellID = 0, Ns = 3, Nf = 101

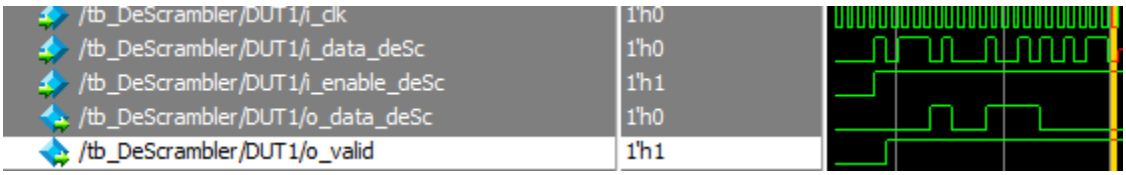
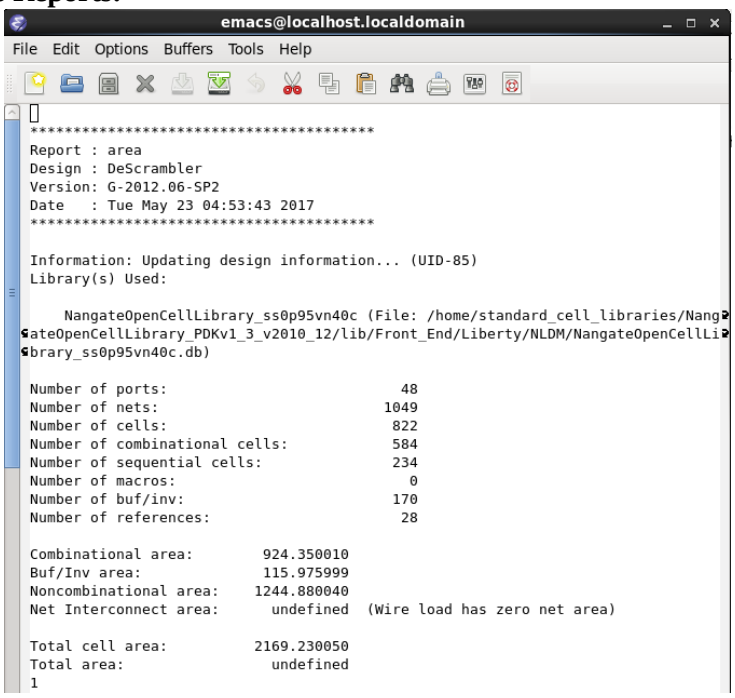


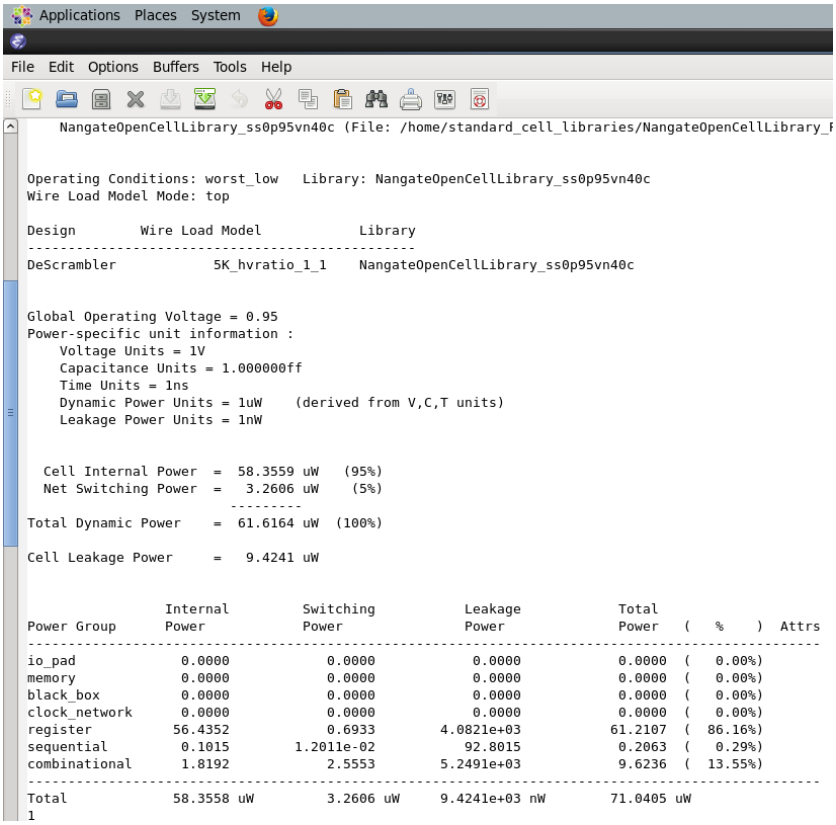
Figure 182 RTL Output from Descrambling for NcellID = 14, Ns = 3, Nf = 101

3.12.5.3. Synthesis Reports:




```
****************************************
Report : area
Design : DeScrambler
Version: G-2012.06-SP2
Date   : Tue May 23 04:53:43 2017
****************************************

Information: Updating design information... (UID-85)
Library(s) Used:

    NangateOpenCellLibrary_ss0p95vn40c (File: /home/standard_cell_libraries/Nang
ateOpenCellLibrary_PDKv1_3_v2010_12/lib/Front_End/Liberty/NLDM/NangateOpenCellLi
brary_ss0p95vn40c.db)

Number of ports:                          48
Number of nets:                         1049
Number of cells:                         822
Number of combinational cells:           584
Number of sequential cells:              234
Number of macros:                          0
Number of buf/inv:                       170
Number of references:                     28

Combinational area:        924.350010
Buf/Inv area:              115.975999
Noncombinational area:    1244.880040
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:          2169.230050
Total area:                 undefined
1
```

Figure 183 Descrambling Area Report




```
    NangateOpenCellLibrary_ss0p95vn40c (File: /home/standard_cell_libraries/NangateOpenCellLibrary_f


Operating Conditions: worst_low   Library: NangateOpenCellLibrary_ss0p95vn40c
Wire Load Model Mode: top

Design        Wire Load Model            Library
------------------------------------------------
DeScrambler            5K_hvratio_1_1    NangateOpenCellLibrary_ss0p95vn40c


Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  58.3559 uW   (95%)
  Net Switching Power  =   3.2606 uW    (5%)
                         ---------
Total Dynamic Power    =  61.6164 uW  (100%)

Cell Leakage Power     =   9.4241 uW


                 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          56.4352            0.6933        4.0821e+03           61.2107  (  86.16%)
sequential         0.1015        1.2011e-02           92.8015            0.2063  (   0.29%)
combinational      1.8192            2.5553        5.2491e+03            9.6236  (  13.55%)
--------------------------------------------------------------------------------------------------
Total             58.3558 uW         3.2606 uW     9.4241e+03 nW        71.0405 uW
1
```

Figure 184 Descrambling Power Report

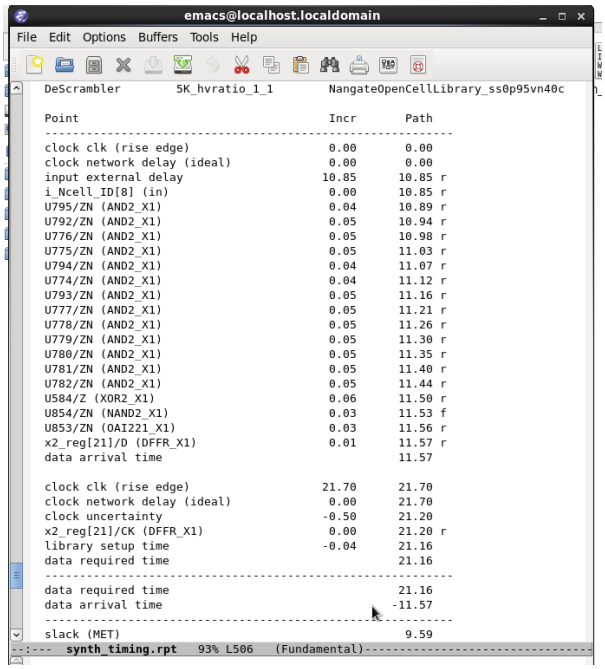


Figure 185 Descrambling Timing Report

3.13. Rate De-Matcher

3.13.1. Block Diagram

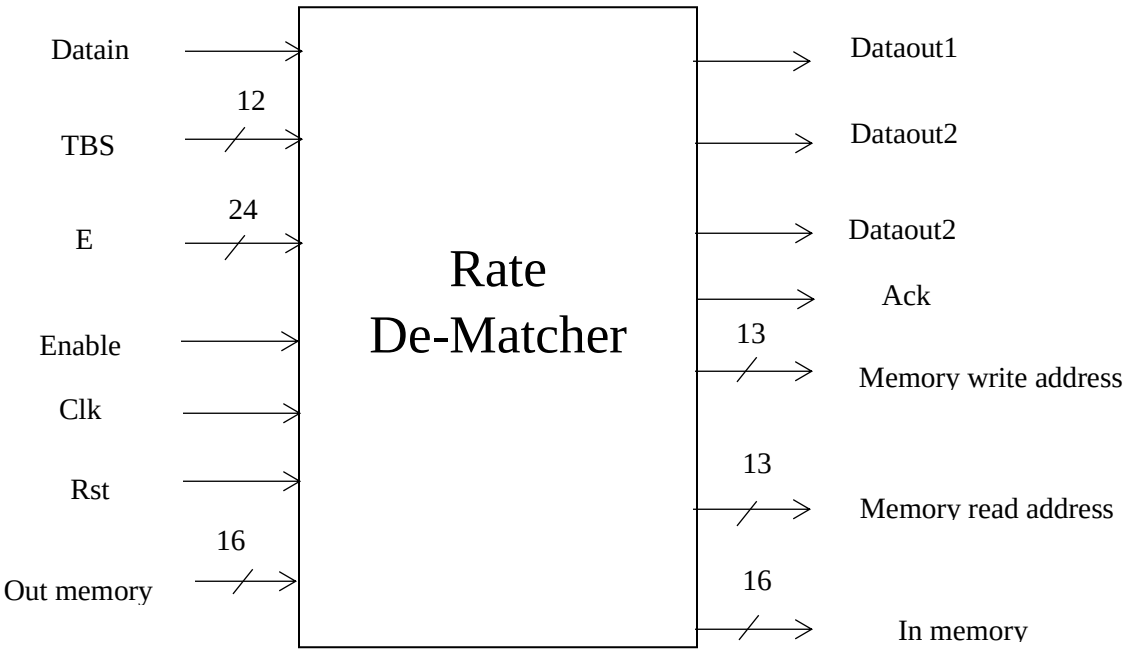


Figure 186: Rate De-Matcher Block Diagram

3.13.1.1. Detailed Block Diagram

3.13.1.1.1. Bit collection block

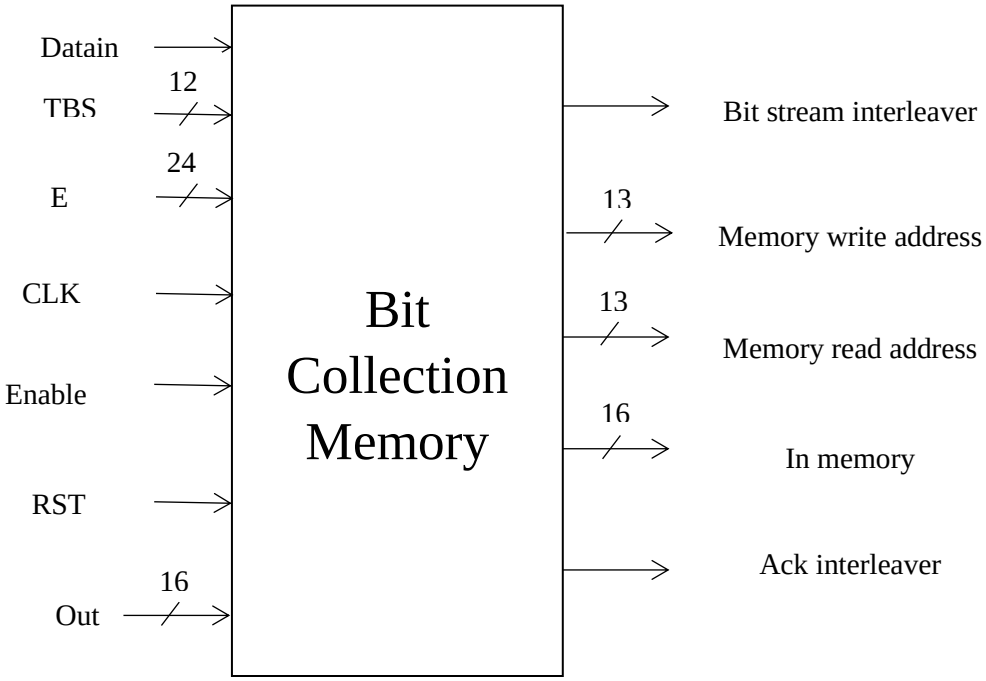


Figure 187: Bit Collection Block Diagram

3.13.1.1.2. De-interleaver block

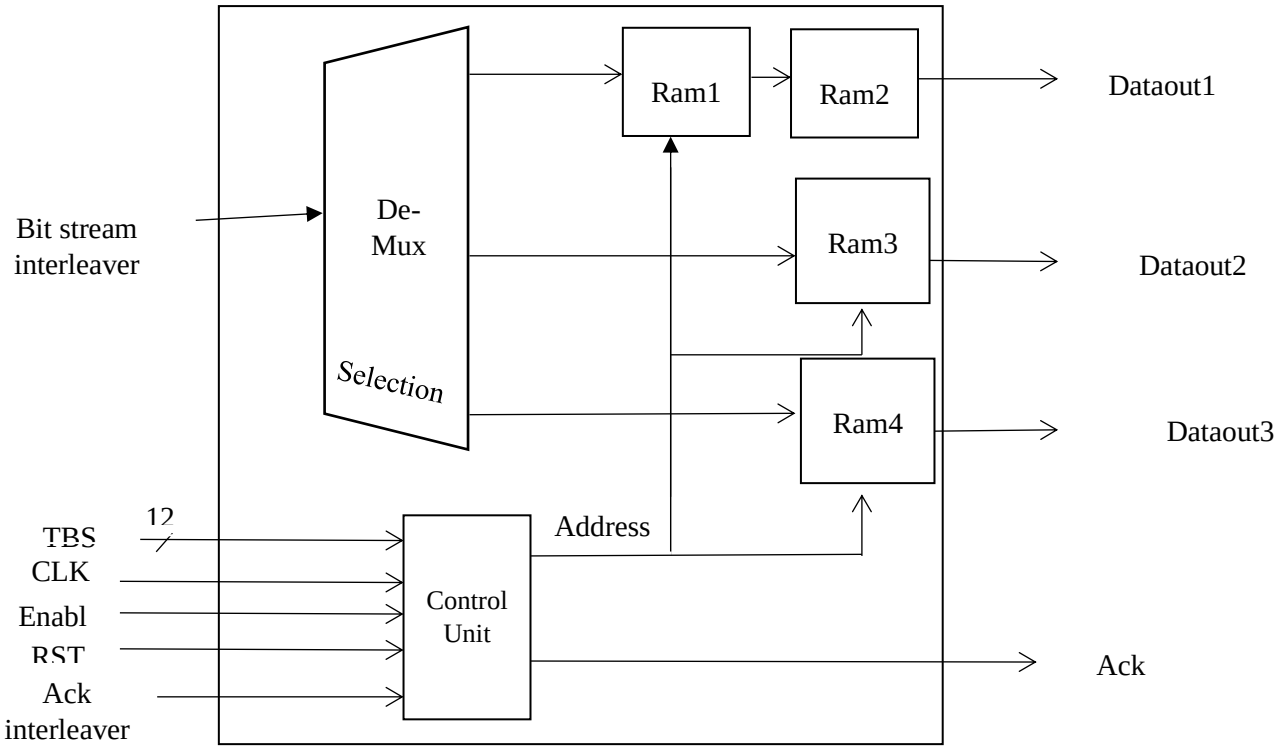


Figure 188 De-interleaver Block Diagram

3.13.2. Interface Table

Table 29 De-Rate Matcher Interface Table

Signal Name	Direction	Width	Description
Datain	Input	1-bit	The input data from Scrambler to the block
TBS (Transport Block Size)	Input	12-bits	The input size of the transport block from the upper layer.
E	Input	24-bits	The size of the data input to the block.
Enable	Input	1-bit	When enable is 1 that means that the input to the block is valid data.
Clk	Input	1-bit	Clock signal.
Rst	Input	1-bit	Reset signal.
Out memory	Input	16-bits	The output data from the shared memory to the block.
Memory write address	Output	13-bits	The write address in the shared memory
Memory read address	Output	13-bits	The read address in the shared memory
In memory	output	16-bits	The input data to the shared memory from the block
Dataout	Output	3-bits	The three bits output to the decoder.
Ack	Output	1-bit	When Ack is 1 that means that the output to the decoder is valid data.

3.13.3. Function of the design [49] [50] [51]

1. Bit collection block:

1- Calculating the virtual circular buffer length by using TBS input
2- Comparing the length of virtual circular buffer with the input data length “E”.
3- If “E” is greater than the VCB, the address pointer will reach the end of VCB then repeat from the beginning and add the data to it.
4- If “E” is equal to the VCB, the address pointer will reach the end of the VCB and will not back to the beginning.
5- If “E” is less than the VCB, the address will reach “E” then continues with zeros till reach the VCB length.
6- After filling this memory, the data will be averaged to the number of repetition (which is calculated from the division of E and the length of the VCB).
7- If this average was smaller than”0.5” the data will be “0”, otherwise it will be “1”.
8- Then the data will be passed to the de-interleaver block.

2. De-interleaver block:

1- Calculating the number of dummy bits as shown in the rate matcher.
2- Calculating the number of rows as shown in the rate matcher.

3- Saving the data in memories column by column according to the permutation table as in the transmitter by taking in consideration the location of dummy bits.
4- After filling the first memory the input data will be saved in the third memory and the data from the first memory will be passed to the second memory at the same time.
5- Then the remained data will be saved in the fourth memory.
6- After finishing filling the memories, the output will be read from them row by row to the decoder by skipping the dummy bits location in each ram.

3.13.4. Design Interface

There are three blocks is communicating with this block:

1- De-scrambler block: This is before this block in NPDSCH.
2- Viterbi Decoder: This is after this block in NPDSCH.
3- Shared memory: The bit collection memory is in this part.

3.13.5. Simulation Results

3.13.5.1. MATLAB Results

There are three cases of MATLAB simulations:

1- TBS = 16 and the input data length E = 60

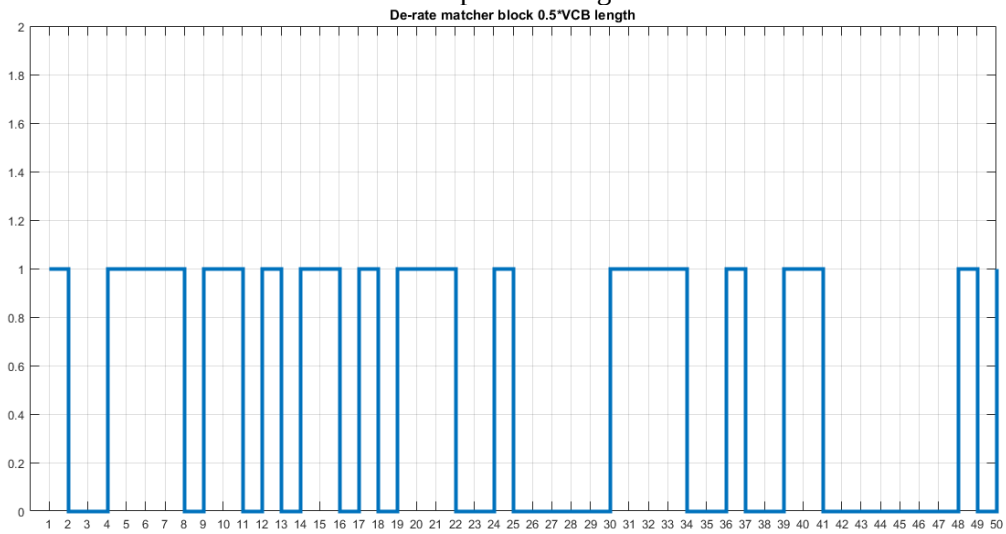


Figure 189Rate De-matcher Output data from MATLAB block for E = 60

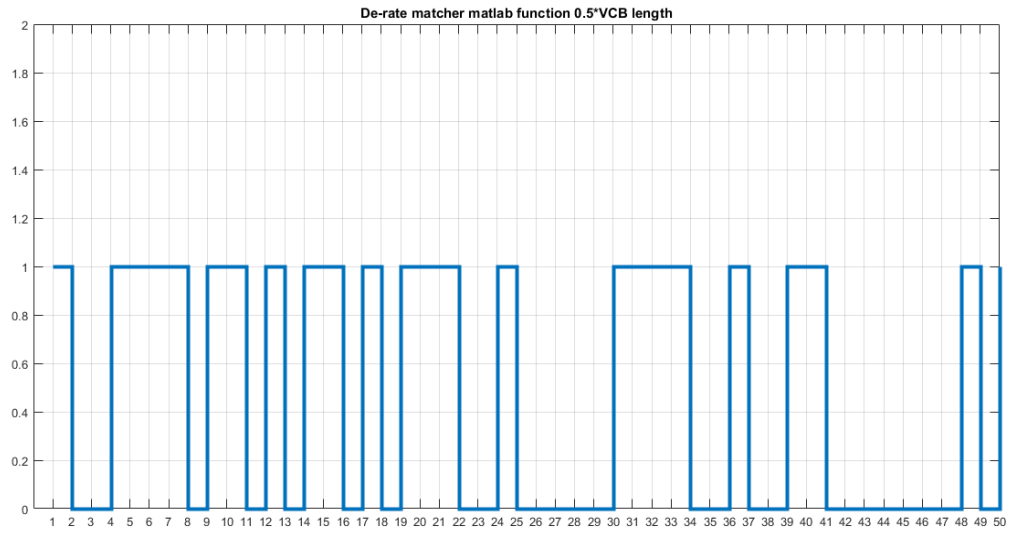


Figure 190:Rate De-matcher Output data from MATLAB function for E = 60

2- TBS = 16 and the input data length E = 120

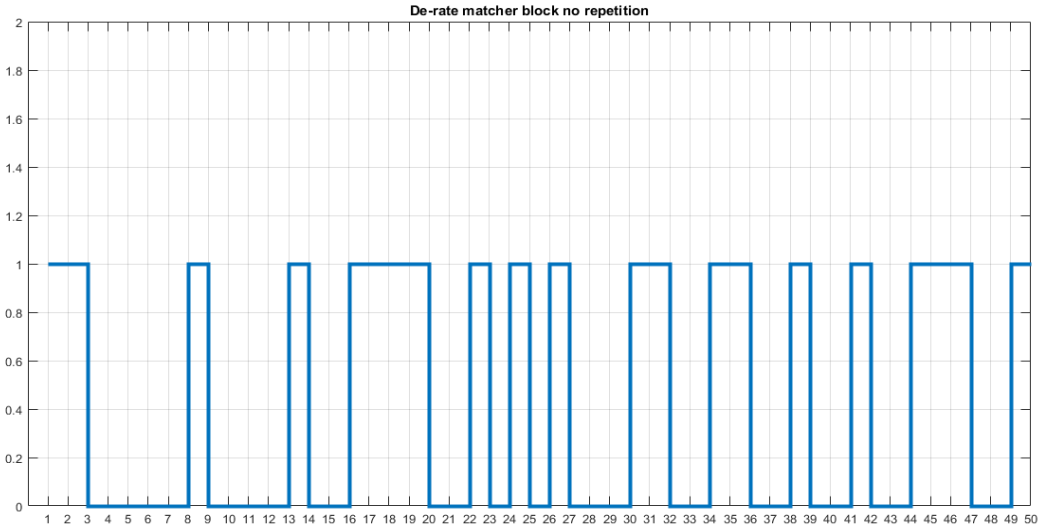


Figure 191:Rate De-matcher Output data from MATLAB block for E = 120

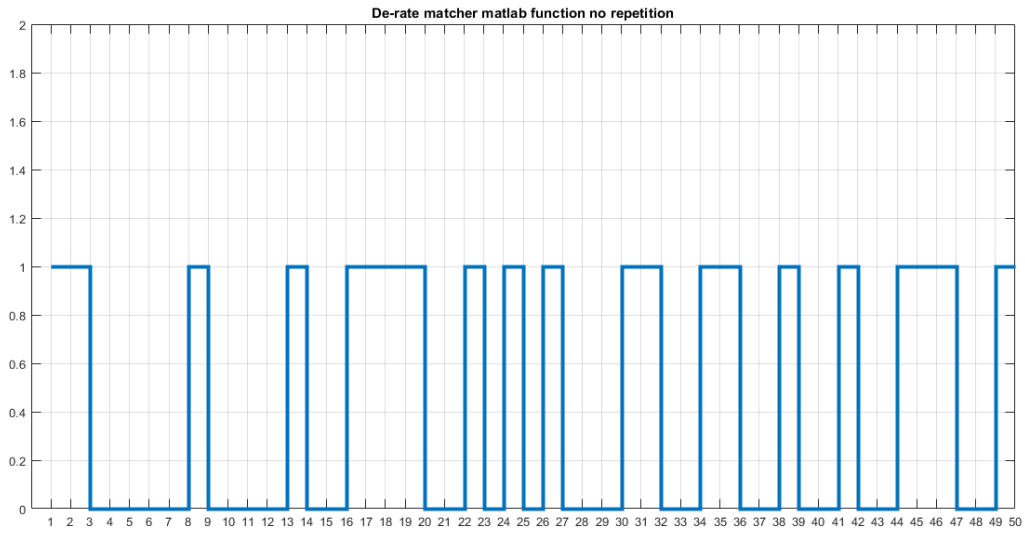


Figure 192 Rate De-matcher Output data from MATLAB function for E = 120

3- TBS = 16 and the input data length E = 240

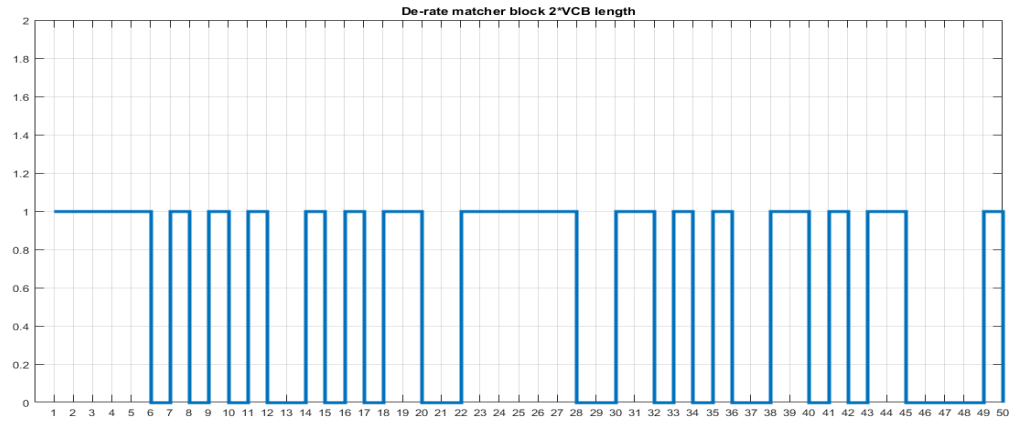


Figure 193 Rate De-matcher Output data from MATLAB block for E = 240

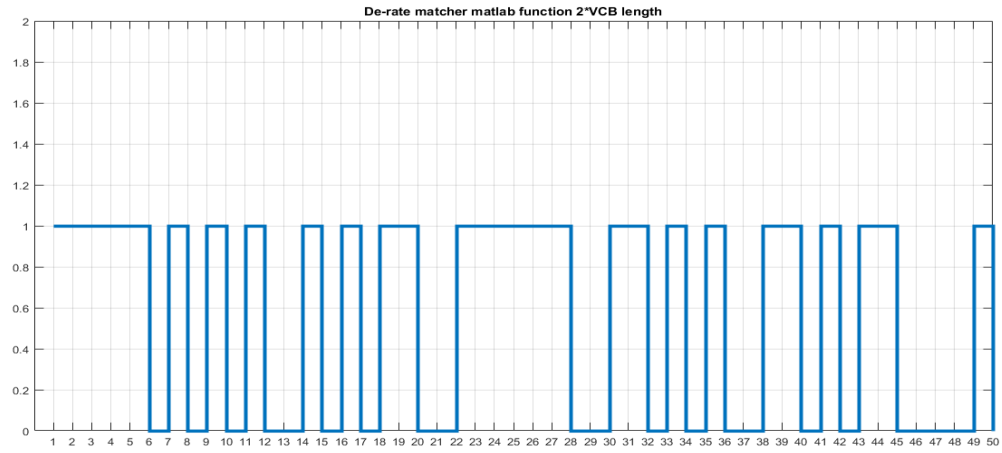


Figure 194 Rate De-matcher Output data from MATLAB function for E = 240

3.13.5.2. RTL Results

RTL result is as same as MATLAB result as shown in the following figures: the first figure is RTL simulation for TBS = 16 and data output length E = 120, Dataout is the three bits output of the block to the decoder. The three figures from MATLAB are determining the three bits output from the block to compare both RTL and MATLAB outputs.

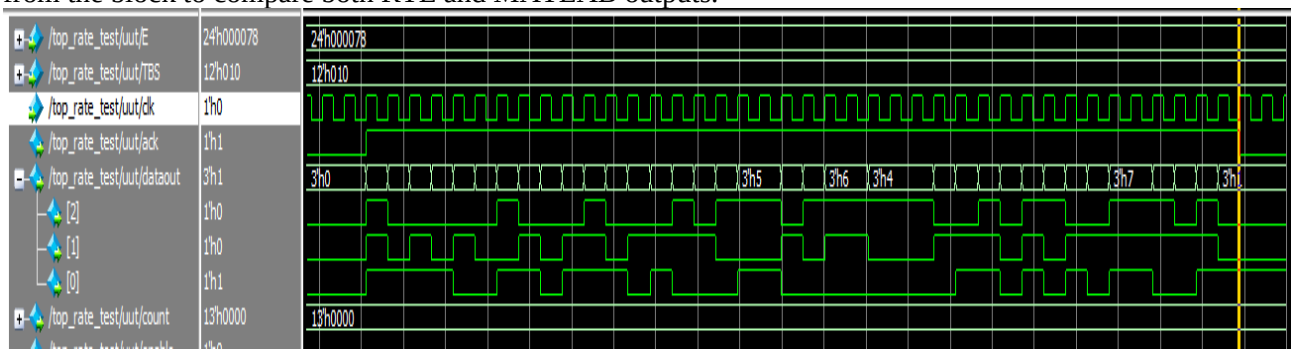


Figure 195 Rate De-matcher RTL output for E = 120

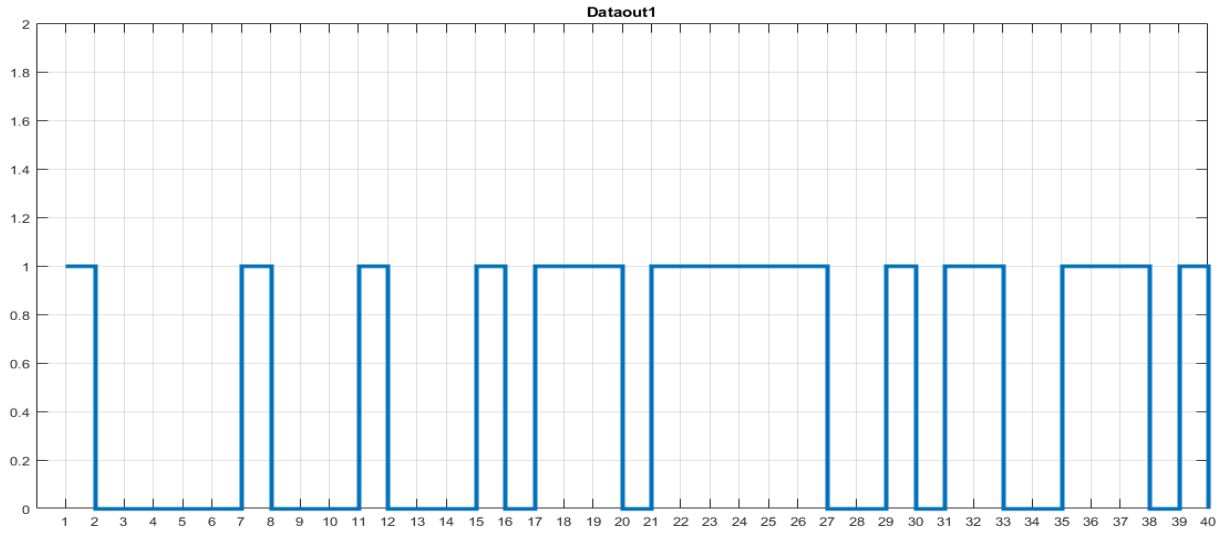


Figure 196 Rate De-matcher MATLAB dataout1

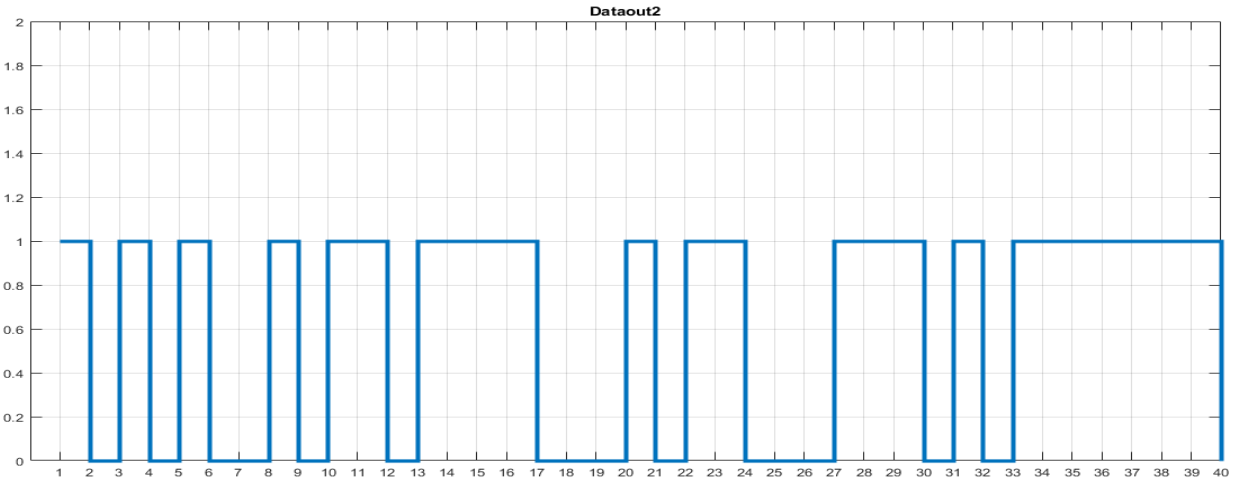


Figure 197 Rate De-matcher MATLAB dataout2

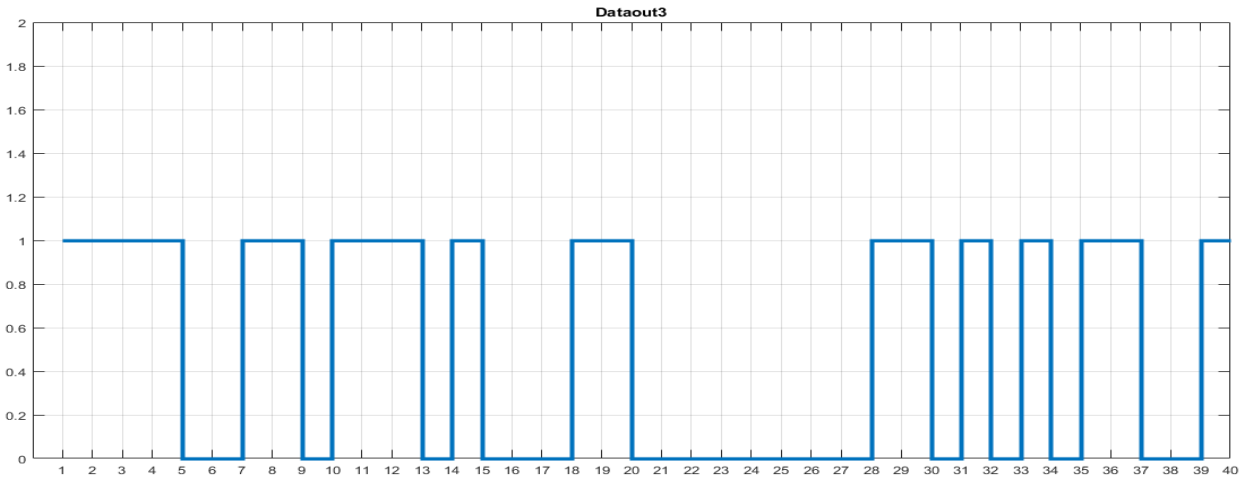


Figure 198 Rate De-matcher MATLAB dataout3

3.13.6. Synthesis Reports

These reports are for the block without the bit collection memory; because it will be a separate memory will be added.

```
Number of ports:                           88
Number of nets:                            88
Number of cells:                            2
Number of combinational cells:              0
Number of sequential cells:                 0
Number of macros:                           0
Number of buf/inv:                          0
Number of references:                       2

Combinational area:       50341.564040
Buf/Inv area:             3828.271920
Noncombinational area:    47289.478352
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:          97631.042392
```

Figure 199 Rate De-matcher Synthesis area report

```
  Cell Internal Power  = 210.8108 uW   (97%)
  Net Switching Power  =   5.9173 uW    (3%)
                       ---------
Total Dynamic Power    = 216.7281 uW  (100%)

Cell Leakage Power     = 414.2407 uW


                 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        1.3668e-02           0.0000          1.3668e-02  (   0.00%)
clock_network  2.8830e-02        1.5955e-02          469.3395            0.5141  (   0.08%)
register         204.6629        2.5053e-02        1.6894e+05          373.6310  (  59.21%)
sequential         0.0000           0.0000            0.0000             0.0000  (   0.00%)
combinational      6.1274           5.8627         2.4486e+05          256.8515  (  40.70%)
--------------------------------------------------------------------------------------------------
Total            210.8191 uW        5.9173 uW     4.1427e+05 nW       631.0103 uW
1
```

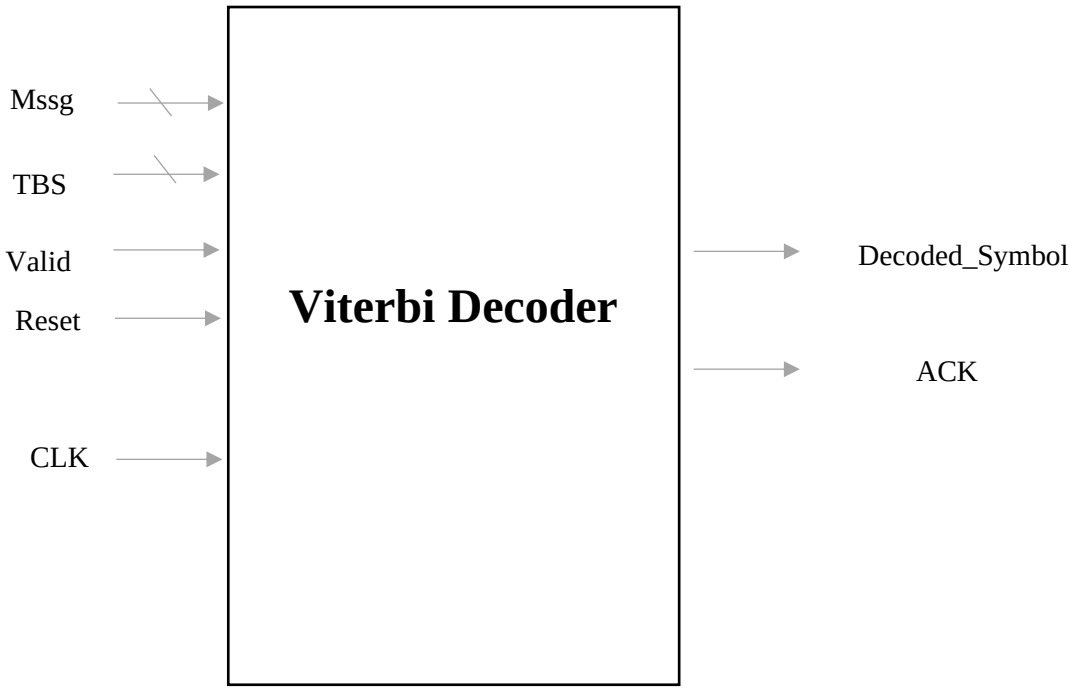
Figure 200 Rate De-matcher Synthesis power report

```
clock clk (rise edge)                    260.00     260.00
clock network delay (ideal)                0.00     260.00
clock uncertainty                         -0.35     259.65
output external delay                     -2.00     257.65
data required time                                  257.65
-----------------------------------------------------------
data required time                                  257.65
data arrival time                                    -2.59
-----------------------------------------------------------
slack (MET)                                         255.06
```

Figure 201 Rate De-matcher Synthesis Timing report

3.14. Viterbi Decoder

3.14.1 Block Diagram:



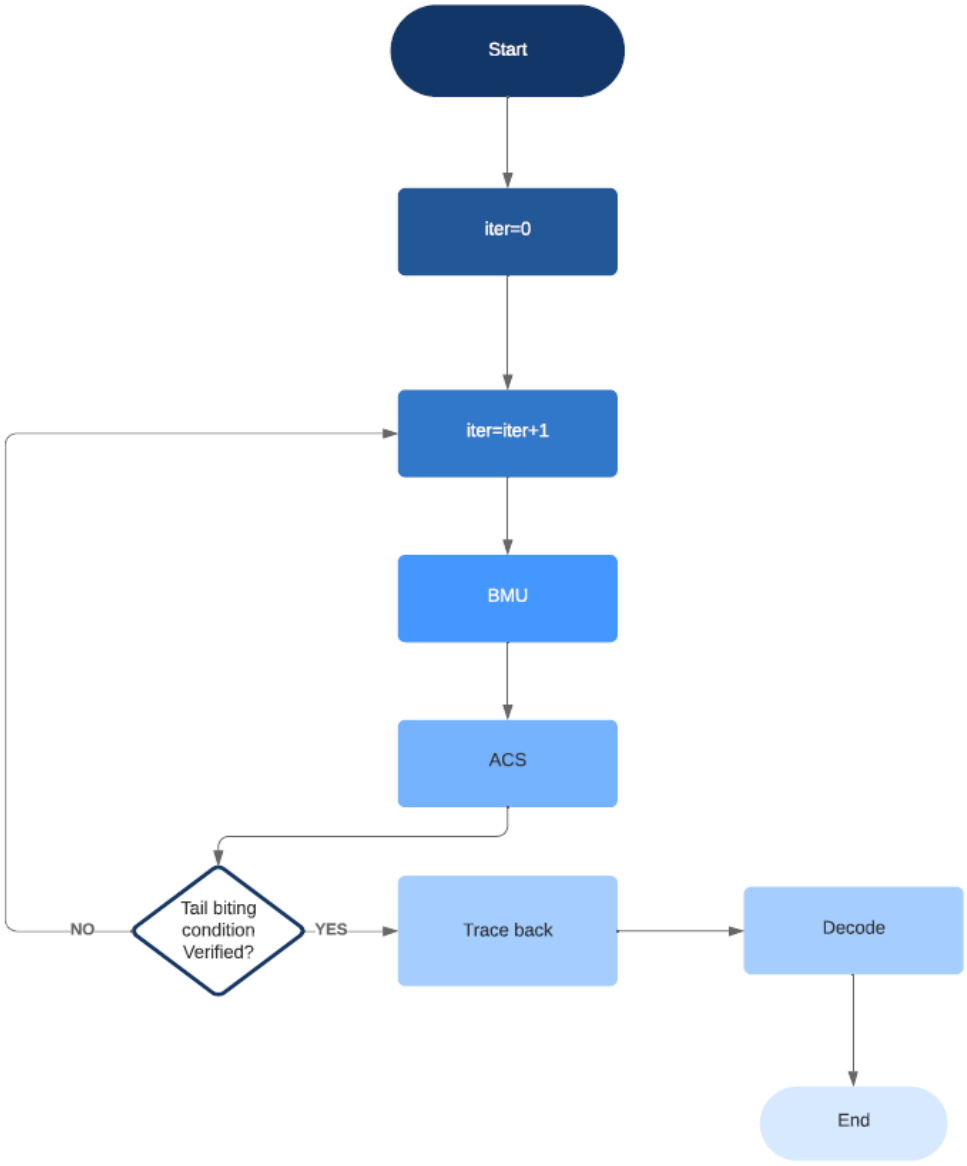
3.14.2 Interface Table:

Table 30: Viterbi Decoder

Signal Name	Direction	Width	Description
Mssg	Input	3	Data stream for code rate of 1/3
TBS	Input	12	Transport Block size (Upper Layer Parameter) Indicating the size of the input bit stream.
CLK	Input	1	System clk
Reset	Input	1	Asynchronous reset
Decoded_Symbol	Output	1	Output bit stream
Ack	Output	1	Acknowledge signal indicating the validity of the output bit stream.

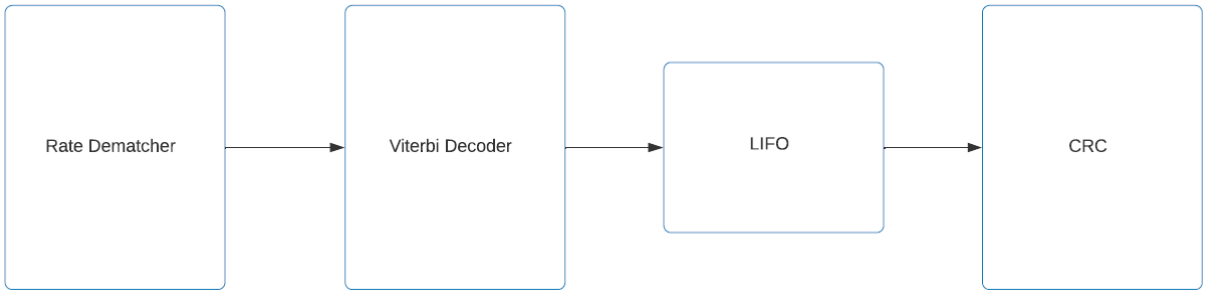
3.14.3 Function of the design:

- It calculates a measure of similarity between the received signal and all the trellis paths entering each state at time.
- Remove all the candidates that are not possible based on the maximum likelihood choice.
- This path is called the surviving path. This selection of surviving paths is done for all the states and makes decisions to eliminate some of the least likely paths in early calculation stages to reduce the decoding complexity.
- The data bit (0 or 1) most likely to have caused entry to each state is stored in a table.
- After a number of clock cycles, defined by the trace back length, the decoder traces back through the trellis, outputting the data bits for the most likely survivor path. This operation is referred to as trace back. Then these bits are passed through a last in, first out (LIFO) structure, so they are output in the order originally received.



3.14.4 Design Interface:

- Input stream is saved in a data memory for multiple iterations before the start of the decoding process.
- Decoded sequence is stored in a LIFO (Last in First out) memory in order for data to be passed in order after the tail biting condition is checked correctly.



3.14.5 Simulation Results:

3.14.5.1 MATLAB Results:

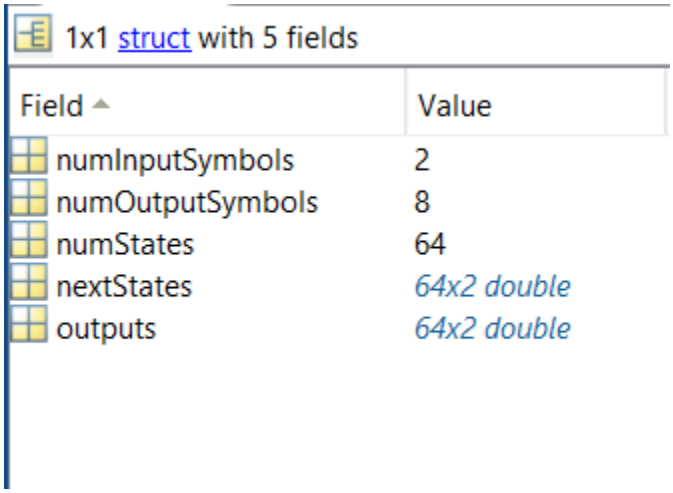


Figure 202: Trellis Structure

trellis.nextStates

	1	2
1	0	32
2	0	32
3	1	33
4	1	33
5	2	34
6	2	34
7	3	35
8	3	35
9	4	36
10	4	36
11	5	37
12	5	37
13	6	38
14	6	38
15	7	39
16	7	39
17	8	40
18	8	40
19	9	41
20	9	41
21	10	42
22	10	42
23	11	43
24	11	43
25	12	44

trellis.nextStates

	1	2
25	12	44
26	12	44
27	13	45
28	13	45
29	14	46
30	14	46
31	15	47
32	15	47
33	16	48
34	16	48
35	17	49
36	17	49
37	18	50
38	18	50
39	19	51
40	19	51
41	20	52
42	20	52
43	21	53
44	21	53
45	22	54
46	22	54
47	23	55
48	23	55
49	24	56

trellis.nextStates

	1	2
50	24	56
51	25	57
52	25	57
53	26	58
54	26	58
55	27	59
56	27	59
57	28	60
58	28	60
59	29	61
60	29	61
61	30	62
62	30	62
63	31	63
64	31	63
65		
66		
67		
68		
69		
70		
71		
72		
73		
74		

Figure 203: Next states in Trellis Diagram for (Rate=1/3, K=7)

trellis.outputs

	1	2
1	0	7
2	7	0
3	4	3
4	3	4
5	1	6
6	6	1
7	5	2
8	2	5
9	6	1
10	1	6
11	2	5
12	5	2
13	7	0
14	0	7
15	3	4
16	4	3
17	7	0
18	0	7
19	3	4
20	4	3
21	6	1
22	1	6
23	2	5
24	5	2
25	1	6

trellis.outputs

	1	2
26	6	1
27	5	2
28	2	5
29	0	7
30	7	0
31	4	3
32	3	4
33	3	4
34	4	3
35	7	0
36	0	7
37	2	5
38	5	2
39	6	1
40	1	6
41	5	2
42	2	5
43	1	6
44	6	1
45	4	3
46	3	4
47	0	7
48	7	0
49	4	3
50	3	4

50	3	4
51	0	7
52	7	0
53	5	2
54	2	5
55	1	6
56	6	1
57	2	5
58	5	2
59	6	1
60	1	6
61	3	4
62	4	3
63	7	0
64	0	7

Figure 204: Trellis Diagram
Expected Outputs

1x64 double

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	951	951	952	951	953	950	954	950	949	950	960	948	951

1x64 double

	14	15	16	17	18	19	20	21	22	23	24	25	26
1	951	950	950	952	950	955	950	951	951	953	951	950	952

1x64 double

	27	28	29	30	31	32	33	34	35	36	37	38	39
1	955	950	952	951	953	951	952	950	955	950	951	950	953

1x64 double

	40	41	42	43	44	45	46	47	48	49	50	51	52
1	951	949	950	957	949	954	952	951	950	953	949	954	953

1x64 double

	53	54	55	56	57	58	59	60	61	62	63	64	65
1	951	951	952	952	949	953	956	949	951	950	954	950	

Figure 205: Path metric Results

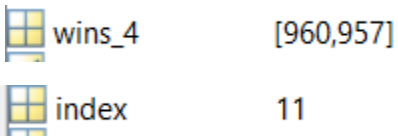
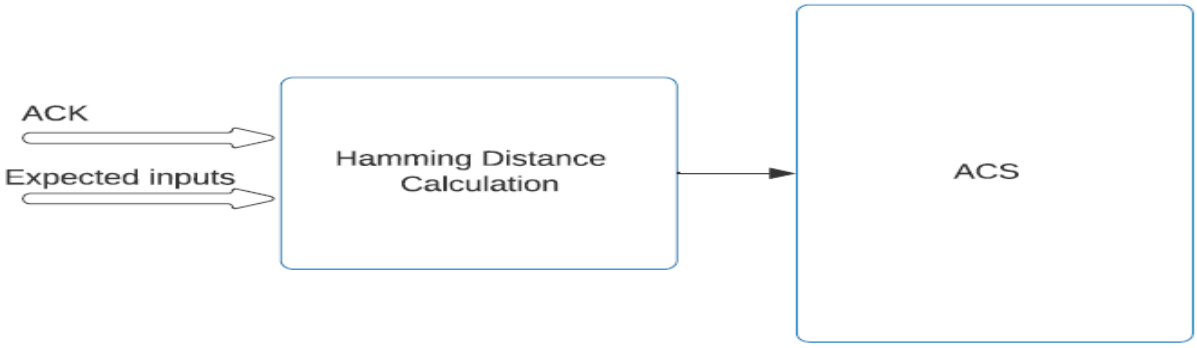


Figure 206: Trace back start point

3.14.5.2. RTL Results:

3.14.1.1. 3.14.5.2.1 BMU Results



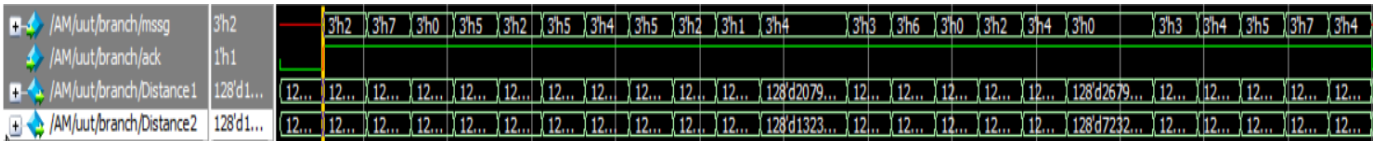


Figure 207: Hamming Distances

3.14.1.2. 3.14.5.2.2 ACS Results

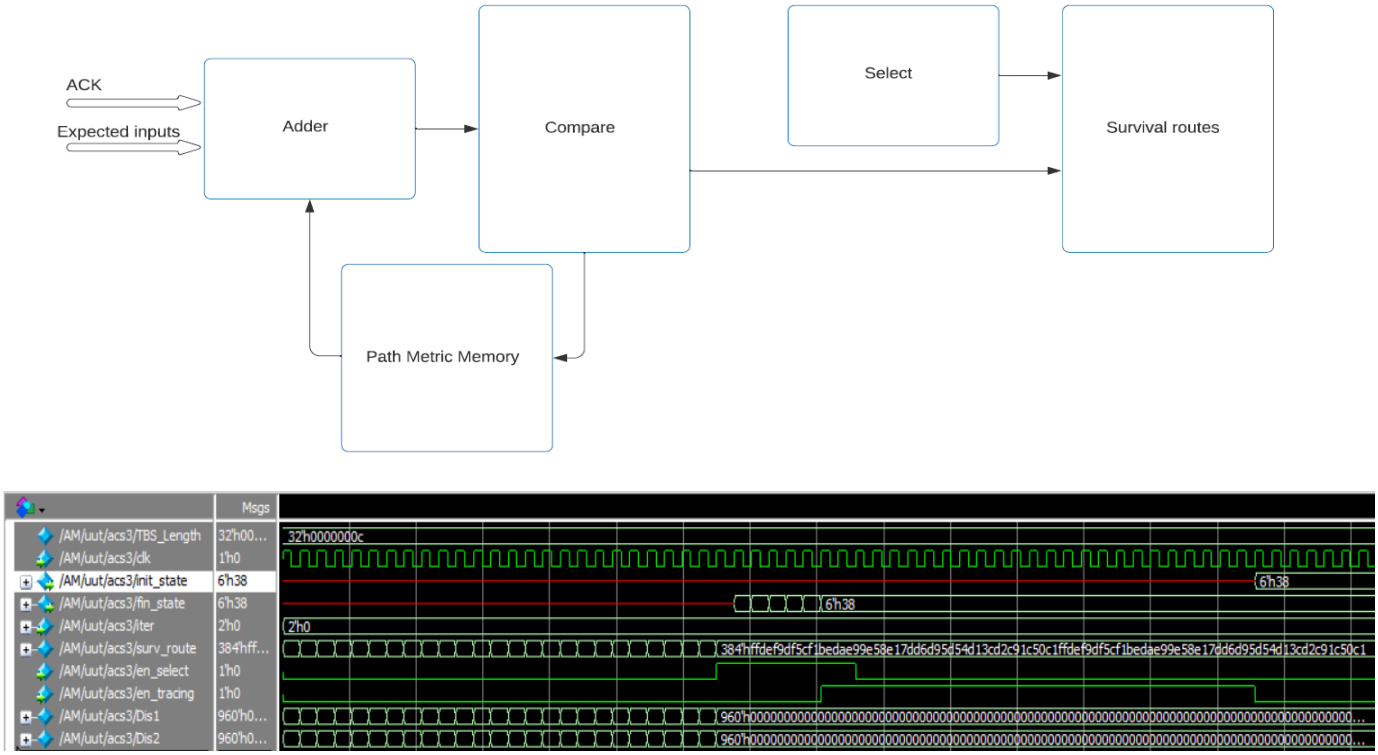


Figure 208: Adder, Compare and select unit results.

3.14.1.3. 3.14.5.2.3 Trace Back Results

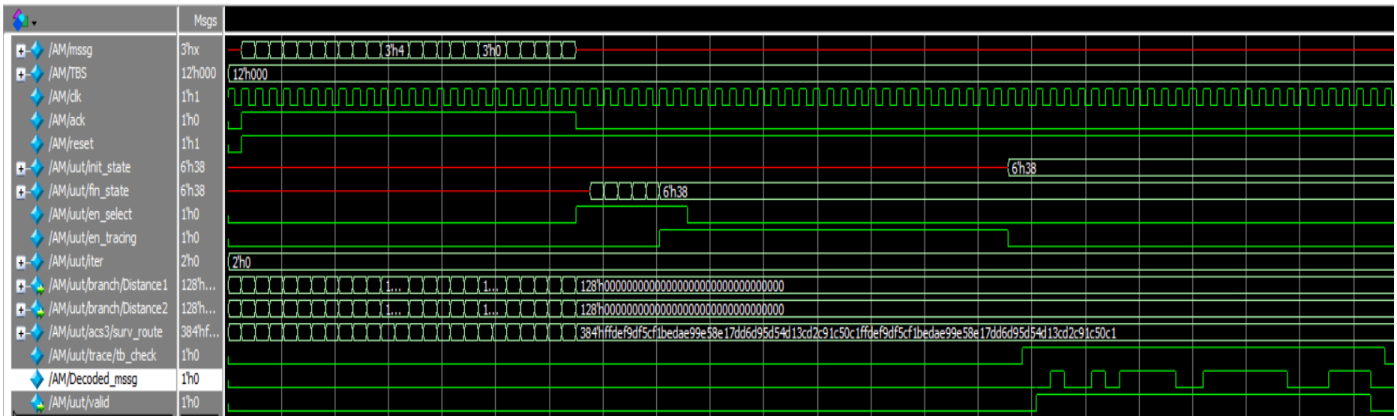


Figure 209: Decoded sequence

3.14.5.3 Synthesis Reports:

```
Library(s) Used:

    NangateOpenCellLibrary_ss0p95vn40c (File: /home/standard_cell_libraries/NangateOpenCellLibrary_PDKv1_3_v2010_12/lib/Front_End/Liberty/NLDM/NangateOpenCellLibrary_ss0p95vn40c.db)


Operating Conditions: worst_low   Library: NangateOpenCellLibrary_ss0p95vn40c
Wire Load Model Mode: top

Design        Wire Load Model            Library
------------------------------------------------
top                   5K_hvratio_1_1    NangateOpenCellLibrary_ss0p95vn40c
```

```
Global Operating Voltage = 0.95
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  =  18.6351 uW   (58%)
  Net Switching Power  =  13.2538 uW   (42%)
                         ---------
Total Dynamic Power    =  31.8889 uW  (100%)

Cell Leakage Power     = 211.1561 uW


                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      3.7218          10.6388        1.2224e+05          136.5988  (  56.20%)
register          14.4063           1.9476        5.8640e+04           74.9937  (  30.86%)
sequential         0.0000           0.0000            0.0000            0.0000  (   0.00%)
combinational      0.5069           0.6672        3.0279e+04           31.4533  (  12.94%)
--------------------------------------------------------------------------------------------------
Total             18.6350 uW        13.2535 uW     2.1116e+05 nW       243.0457 uW
1
```

Figure 210: Power Report

```
Number of ports:                          21
Number of nets:                          553
Number of cells:                         222
Number of combinational cells:           189
Number of sequential cells:               28
Number of macros:                          0
Number of buf/inv:                        37
Number of references:                     24

Combinational area:       26714.646108
Buf/Inv area:             3469.172016
Noncombinational area:    17277.232419
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:          43991.878527
Total area:                 undefined
```

Figure 211: Area Report

```
clock clk (rise edge)                    520.00     520.00
clock network delay (ideal)                0.00     520.00
clock uncertainty                         -0.50     519.50
acs3/ram/rd_surv_reg[0]/CK (DFF_X1)        0.00     519.50 r
library setup time                        -0.03     519.47
data required time                                  519.47
-----------------------------------------------------------
data required time                                  519.47
data arrival time                                  -103.85
-----------------------------------------------------------
slack (MET)                                         415.61
```

Figure 212: Timing Report

3.15. Cyclic Redundancy Check

3.15.1. Block Diagram

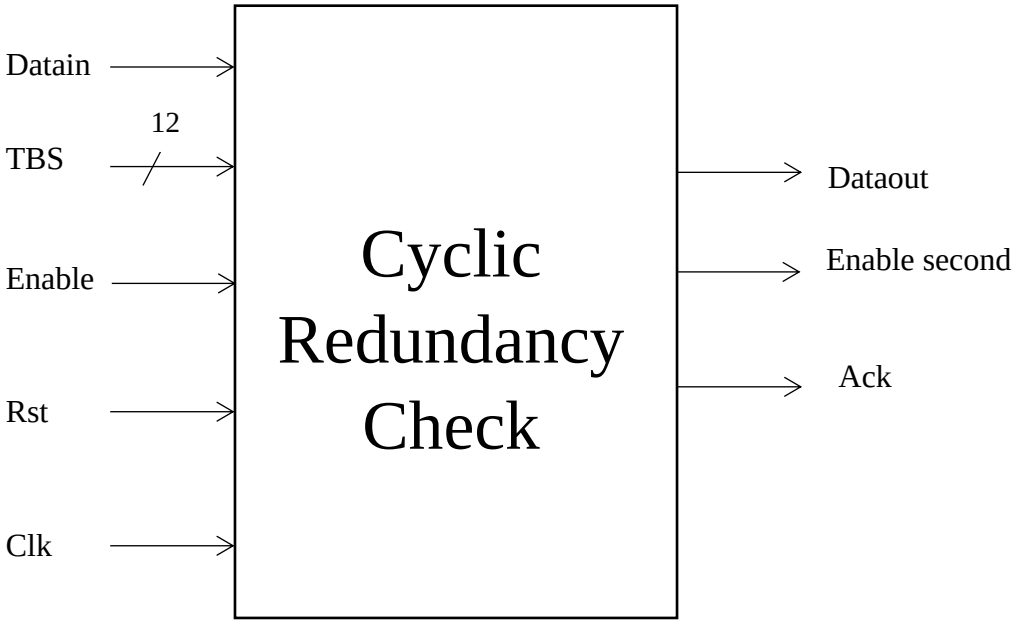


Figure 213 CRC Block Diagram

3.15.1.1. Detailed Block Diagram

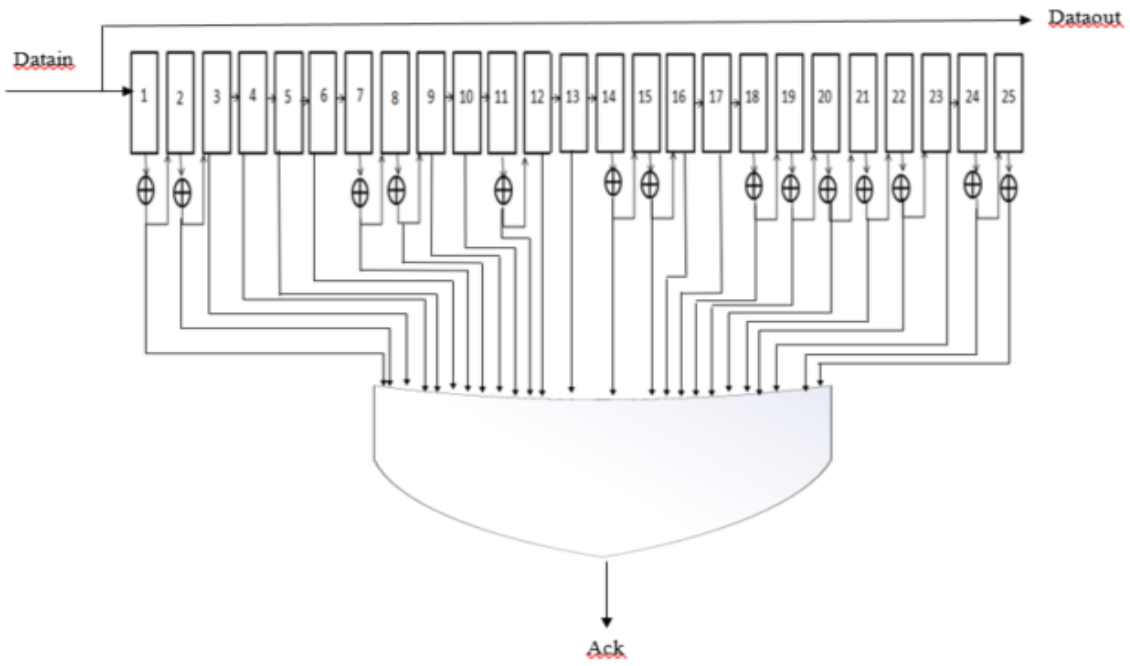


Figure 214: CRC Detailed Block Diagram

3.15.2. Interface Table:

Table 31: CRC Interface Table

Signal Name	Direction	Width	Description
Datain	Input	1-bit	Input data to the block.
TBS	Input	12-bits	Transport Block Size.
Enable	Input	1-bit	When enable is 1 that means that the input to the block is valid data.
Rst	Input	1-bit	Reset signal
Clk	Input	1-bit	Clock signal
Dataout	Output	1-bit	Output data from the block.
Enable second	Output	1-bit	When Enable second is 1 that means that the output to the decoder is valid data.
Ack	Output	1-bit	When Ack is 1 that means that the data is correct, otherwise the data is wrong.

3.15.3. Function of the design:

1- The input enters this block serial into the 25-shift register.
2- When it reaches the most significant bit, it checks if it is 1 or not.
3- If it was 0, it shifts the input bits to find the first 1 in the data.
4- If it was 1, XOR operation will start with the polynomial.
5- It continues this XOR operation until the data ends.
6- Then checks the last 24 bits if they are 0, that means the data is correct, otherwise the data is wrong.

3.15.4. Design Interface:

This block is communicating with:

1- The Viterbi decoder.
2- The upper layer.

3.15.5. Simulation Results:

3.15.5.1. MATLAB Results:

This case when TBS = 32, Input data length = 56.

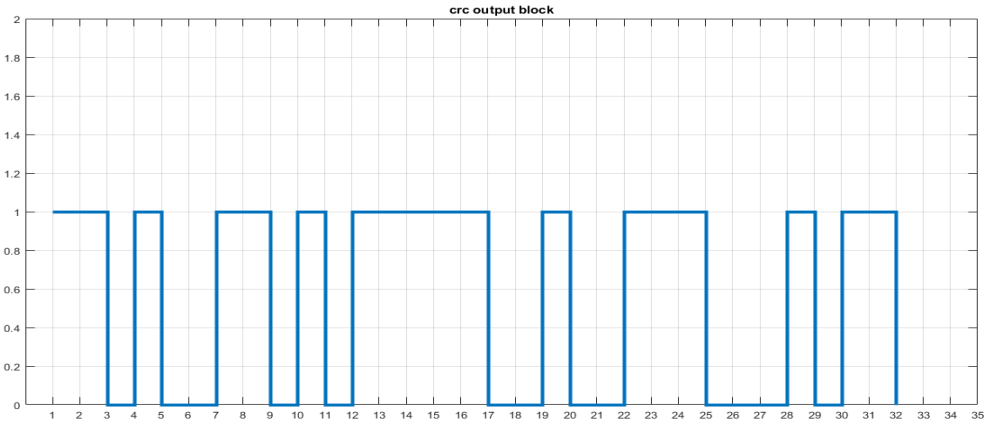


Figure 215: output CRC block

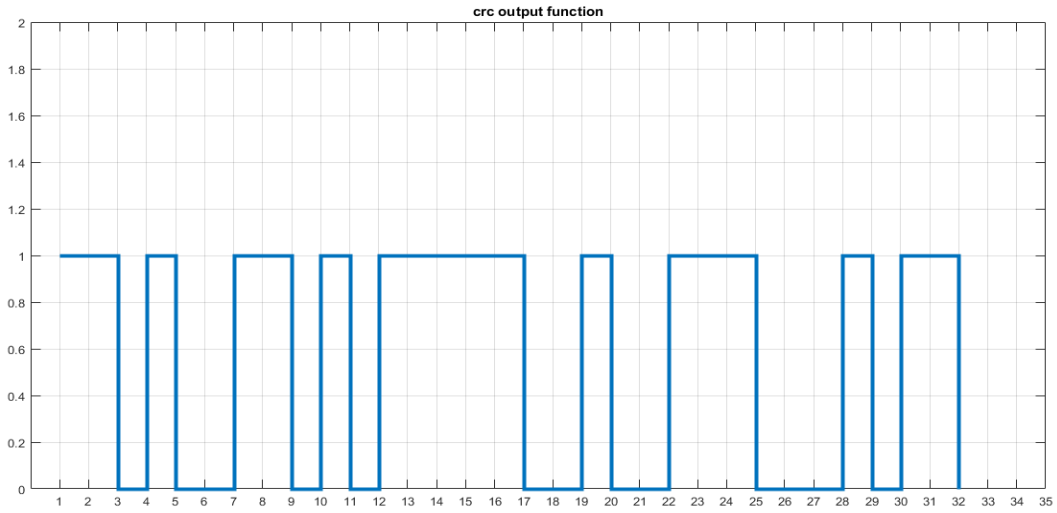


Figure 216: output CRC function

3.15.5.2. RTL Results:

This case when TBS = 32, Input data length = 56.

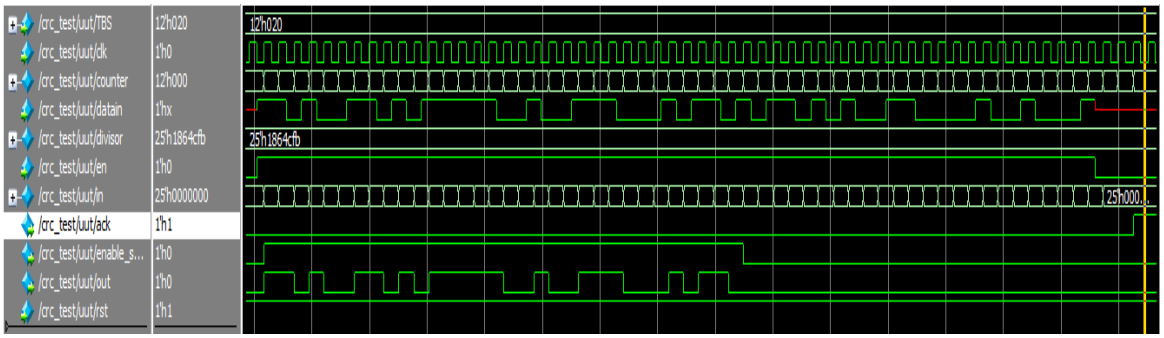


Figure 217: output CRC RTL

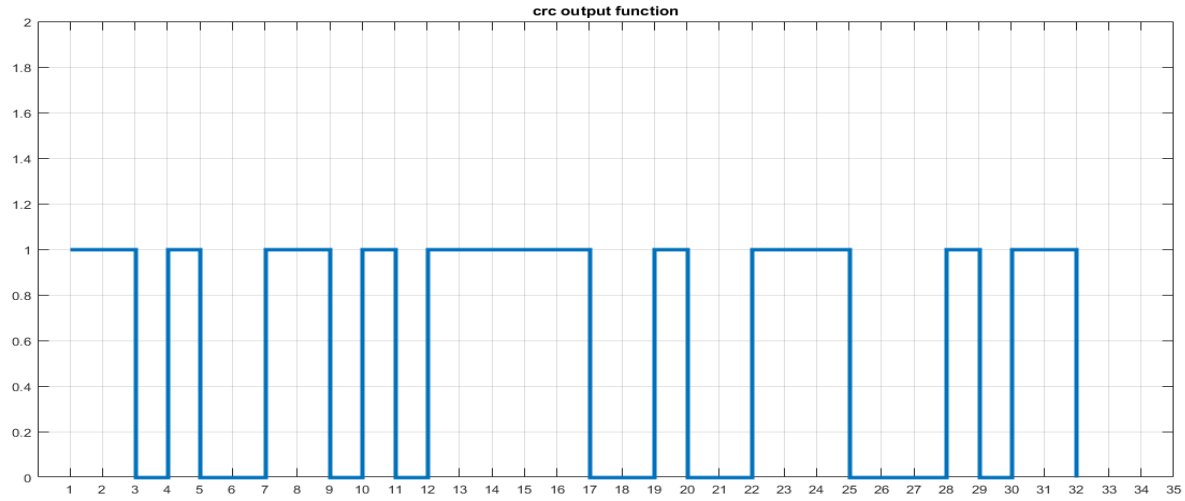


Figure 218: output CRC function MATLAB

3.15.5.3. Synthesis Reports:

```
  Cell Internal Power  = 896.3353 nW   (90%)
  Net Switching Power  = 104.0971 nW   (10%)
                         ---------
Total Dynamic Power    =   1.0004 uW  (100%)

Cell Leakage Power     =   2.0690 uW


                 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           0.8070        1.9674e-02          719.3258            1.5460  (  50.37%)
sequential         0.0000           0.0000            0.0000            0.0000  (   0.00%)
combinational  8.9374e-02        8.4424e-02        1.3497e+03            1.5235  (  49.63%)
--------------------------------------------------------------------------------------------------
Total              0.8963 uW         0.1041 uW     2.0690e+03 nW         3.0695 uW
```

Figure 219: CRC Synthesis power report

```
clock clk (rise edge)                          260.00      260.00
clock network delay (ideal)                      0.00      260.00
clock uncertainty                               -0.35      259.65
counter_reg[3]/CK (DFFR_X1)                      0.00      259.65 r
library setup time                              -0.04      259.61
data required time                                         259.61
-----------------------------------------------------------------
data required time                                         259.61
data arrival time                                           -3.24
-----------------------------------------------------------------
slack (MET)                                                256.37
```

Figure 220: CRC Synthesis timing report

```
Number of ports:                           19
Number of nets:                           274
Number of cells:                          229
Number of combinational cells:            188
Number of sequential cells:                40
Number of macros:                           0
Number of buf/inv:                         49
Number of references:                      25

Combinational area:        244.720000
Buf/Inv area:               27.132000
Noncombinational area:     212.800007
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:           457.520007
Total area:                 undefined
```

Figure 221: CRC Synthesis area report

3.16. RAM Sharing

3.16.1. Block Diagram:

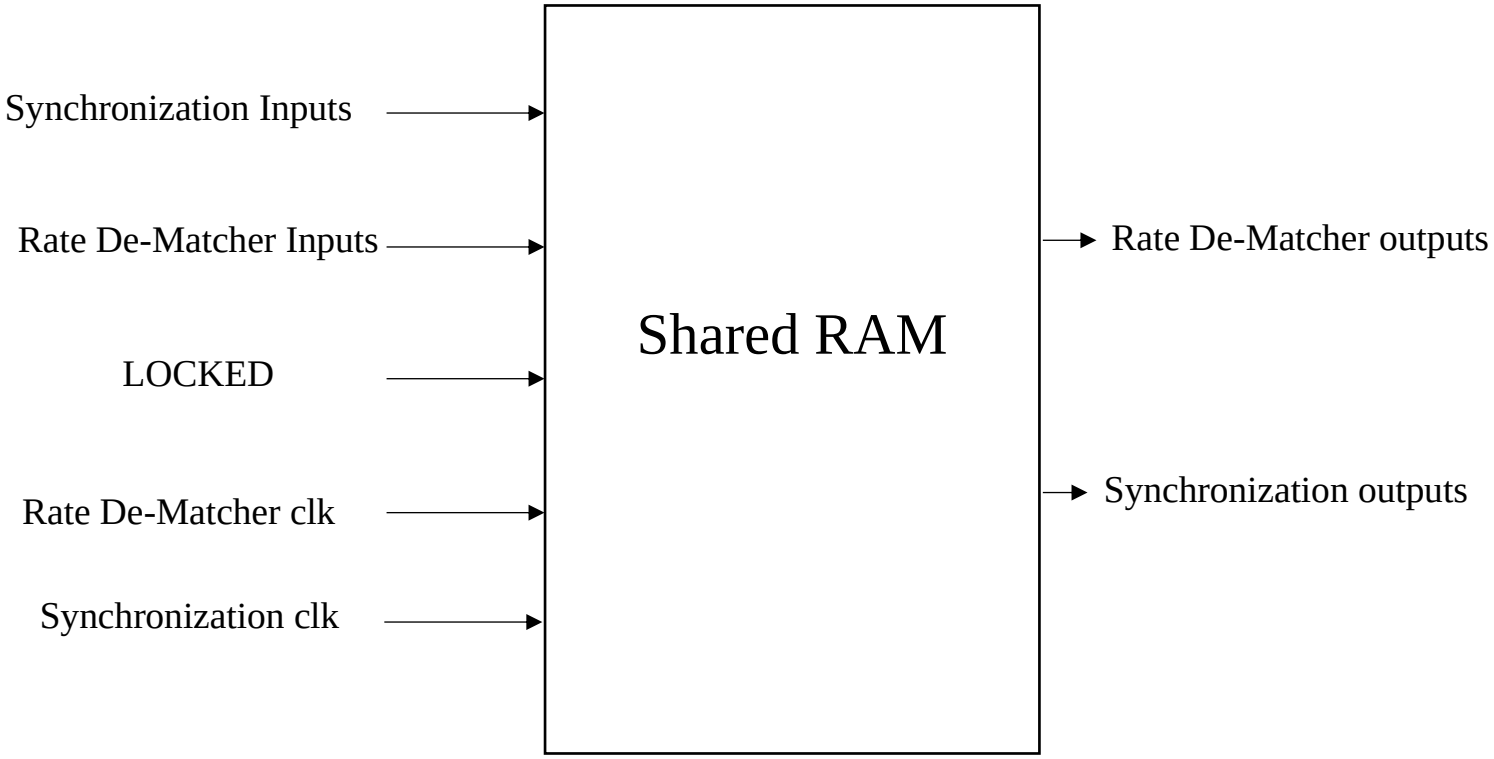


Figure 222 Shared RAM Block Diagram

3.16.2. Interface Table:

Table 32 Shared RAM Interface Table

Direction	Signal Name	Width	Description
Synchronization Inputs	Ram0 Enable	1	Ram0 write enable level-sensitive signal
	Real Ram1 Enable	1	Real ram1 write enable level-sensitive signal
	Imaginary Ram1 Enable	1	Imaginary ram1 write enable level-sensitive signal
	Real Ram2 Enable	1	Real ram2 write enable level-sensitive signal
	Imaginary Ram2 Enable	1	Imaginary ram2 write enable level-sensitive signal
	Real Ram3 Enable	1	Real ram3 write enable level-sensitive signal
	Imaginary Ram3 Enable	1	Imaginary ram3 write enable level-sensitive signal
	Ram4 Enable	1	Real&Imaginary ram4 write enable level-sensitive signal
	Ram5 Enable	1	Real&Imaginary ram5 write enable level-sensitive signal
	Real Ram6 Enable	1	Real ram6 enable level-sensitive signal
	Imaginary Ram6 Enable	1	Imaginary ram6 write enable level-sensitive signal
	Ram7 Enable	1	Real&Imaginary ram7 write enable level-sensitive signal
	Ram0 Address	8	Ram0 read&write address
	Ram1 Address	9	Real&Imaginary ram1 read&write address
	Ram2 Address	9	Real&Imaginary ram2 read&write address
	Ram3 Address	8	Real&Imaginary ram3 read&write address
	Ram4 Address	8	Real&Imaginary ram4 read&write address
	Ram5 Address	8	Real&Imaginary ram5 read&write address
	Ram6 Address	9	Real&Imaginary ram6 read&write address
	Write Ram7 Address	11	Real&Imaginary ram7 write address
	Read Ram7 Address	11	Real&Imaginary ram7 read address
	Ram0 Write Data	16	Ram0 input data
	Real Ram1 Write Data	16	Real ram1 input data
	Imaginary Ram1 Write Data	16	Imaginary ram1 input data
	Real Ram2 Write Data	16	Real ram2 input data
	Imaginary Ram2 Write Data	16	Imaginary ram2 input data
	Real Ram3 Write Data	16	Real ram3 input data
	Imaginary Ram3 Write Data	16	Imaginary ram3 input data
	Real Ram4 Write Data	16	Real ram4 input data
	Imaginary Ram4 Write Data	16	Imaginary ram4 input data
	Real Ram5 Write Data	16	Real ram5 input data
	Imaginary Ram5 Write Data	16	Imaginary ram5 input data
	Real Ram6 Write Data	16	Real ram6 input data

	Imaginary Ram6 Write Data	16	Imaginary ram6 input data
	Real Ram7 Write Data	16	Real ram7 input data
	Imaginary Ram7 Write Data	16	Imaginary ram7 input data
Synchronization outputs	Ram0 Read Data	16	Ram0 output data
	Real Ram1 Read Data	16	Real ram1 output data
	Imaginary Ram1 Read Data	16	Imaginary ram1 output data
	Real Ram2 Read Data	16	Real ram2 output data
	Imaginary Ram2 Read Data	16	Imaginary ram2 output data
	Real Ram3 Read Data	16	Real ram3 output data
	Imaginary Ram3 Read Data	16	Imaginary ram3 output data
	Real Ram4 Read Data	16	Real ram4 output data
	Imaginary Ram4 Read Data	16	Imaginary ram4 output data
	Real Ram5 Read Data	16	Real ram5 output data
	Imaginary Ram5 Read Data	16	Imaginary ram5 output data
	Real Ram6 Read Data	16	Real ram6 output data
	Imaginary Ram6 Read Data	16	Imaginary ram6 output data
	Real Ram7 Read Data	16	Real ram7 output data
	Imaginary Ram7 Read Data	16	Imaginary ram7 output data
Rate De-Matcher Output	Out memory	16	The output data from the shared memory to the block.
Rate De-Matcher Inputs	Memory write address	13	The write address in the shared memory
	Memory read address	13	The read address in the shared memory
	In memory	16	The input data to the shared memory from the block
Sharing RAM Clocks	Rate De-Matcher clk	1	Rate De-Matcher clock 3.846 MHz
	Synchronization clk	1	Coarse Synchronization Clock 1.92 MHz
Multiplexing Signal	LOCKED	1	A level sensitive signal that indicates the success and end of the synchronization procedure

Chapter 4
System Integration and Results

4.1. MATLAB integration:

After finishing MATLAB model for block level, a NPDSCH transmitter is designed using MATLAB built-in functions based on 3GPP standard Release 14. Then we added the channel model ETA with maximum Doppler shift of 5 Hz, 9 taps, and maximum excess tap delay of 5 us. Then, we added a noise model using AWGN with minimum required SNR $-12.6\ dB$ for the design. Finally, we added frequency and time offset, the range of frequency offset is $[-35:35]\ KHz$ and time offset range $[0:192000]$ samples.

After transmitter and channel modeling, the receiver chain is added in a generic format to satisfy all TBS and repetition ranges.

The system is tested for repetitions $[1, 32, 64, 128, 256, 512, 1024]$ under SNR ranges $[-14:20]dB$ and the result of the testing is as shown in Figure 223 indicates to Bit Error Rate and Figure 224 indicates to Block Error Rate.

The results shown are compared with MATLAB model results as in Figure 225. The deference between our model and the MATLAB model that cause this deference in the graphs are:

- MATLAB model uses a perfect synchronization.
- MATLAB model uses a perfect frequency correction by using exponential as we use cordic in our chain.
- MATLAB model uses a perfect channel estimation.
- MATLAB model uses a soft symbol de-mapper
- MATLAB model uses a soft decoding.

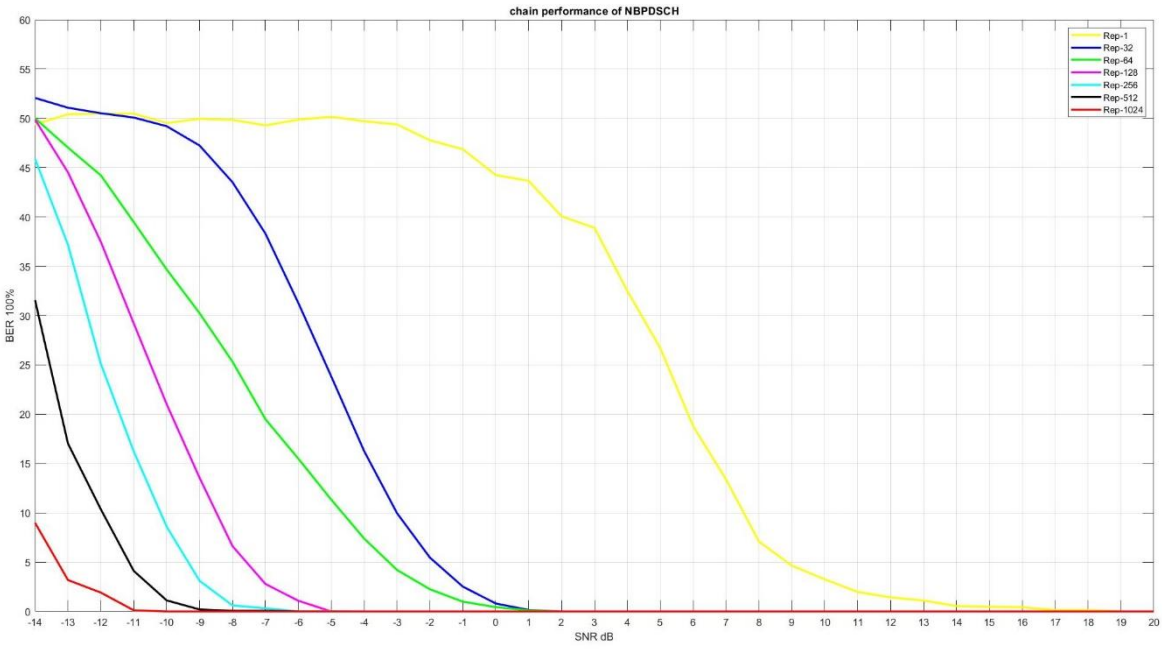


Figure 223 MATLAB BER Vs SNR

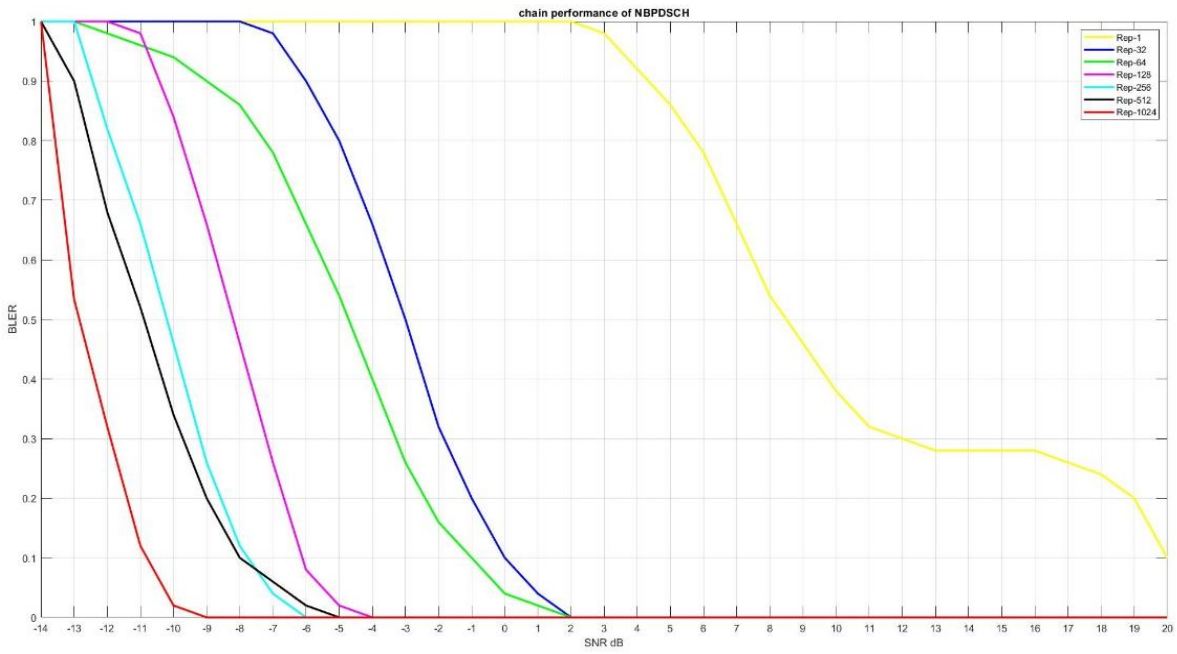


Figure 224 MATLAB BLER Vs SNR

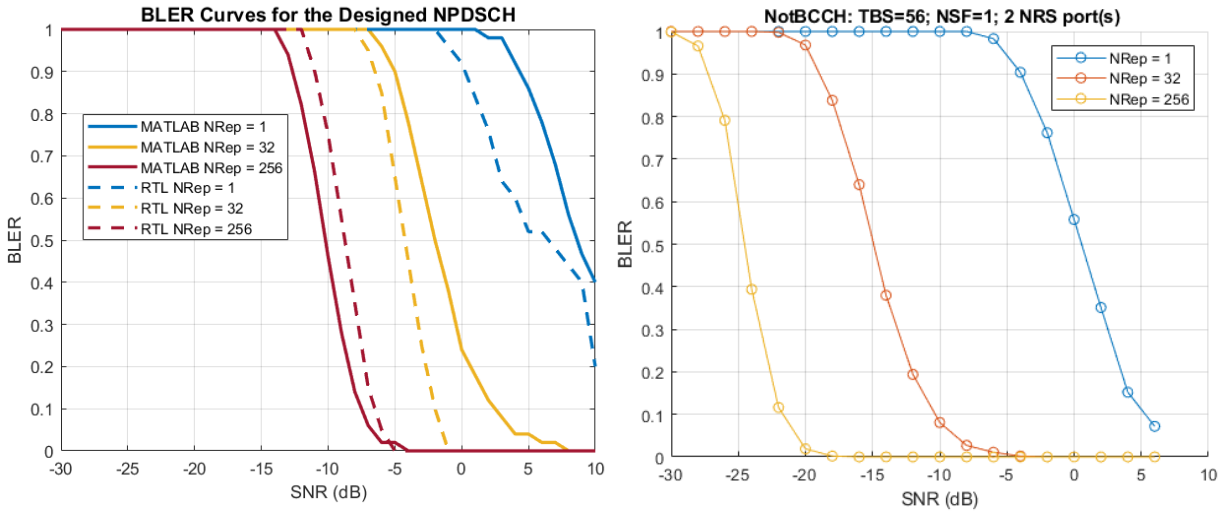


Figure 225 BLER Vs SNR for MATLAB and RTL designed model compared with ideal model

4.2.RTL integration:

In rates and specification phase we settle on the signal interface for each block and the out rate from each block that help us in chain integration.

Integration of the chain is done in 3 parallel levels:

- coarse synchronization, CFO, FFT, and resource de-mapper together.
- Fine Synchronization, Channel Estimation, Channel Equalization, symbol de-mapper, and descrambler together.
- rate de-matcher, decoding, and CRC together.

By using MATLAB to generate a text file and we make the test bench read it symbol per clock cycle. After making sure that each group works correctly, we integrated the 3 parts together.

The RTL System is tested with the same way that used in MATLAB, a python script is used to run MATLAB to generate the files, then run ModelSIM to simulate the design, finally the running MATLAB again to calculate BER and BLER and the results of the them are as shown in figures 208, 209 respectively.

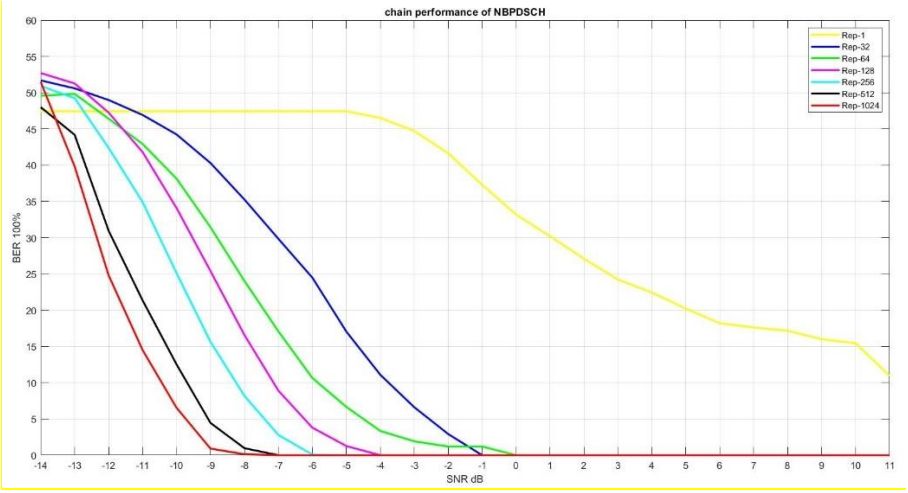


Figure 226 RTL BER Vs SNR

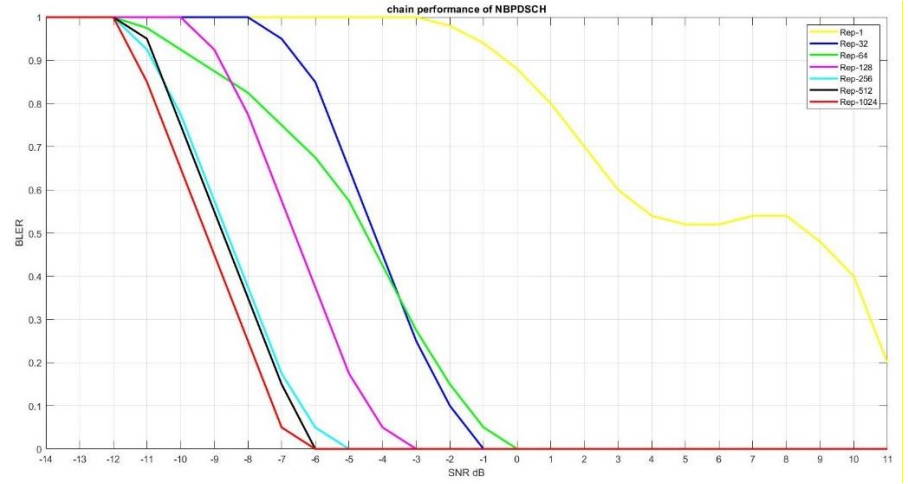


Figure 227 RTL BLER Vs SNR

4.3.Synthesis:

The whole design is synthesized using two tools Synopsys Design Compiler 45 nm and Xilinix VIVADO 28 nm.

4.3.1. DC results:

Area, power, and latency comparison between all blocks of our final integrated chain is shown in the following graphs

Area (um2)	
CSYNCH	109445.17
CFO	2470.874
FFT	22890.896
R De-Mapper	57214.205
CH EST	19910.898
NRS Generation	3439.114
NRS Location	501.144
CH Equalizer	63842.926
Fine synchronization	12009.634
NRS Removal	3950.632
De-mapper	25.8
De-scrambler	2196.23
Rate De-Matcher	97631.042
Channel Decoder	43991.878
CRC	457.52

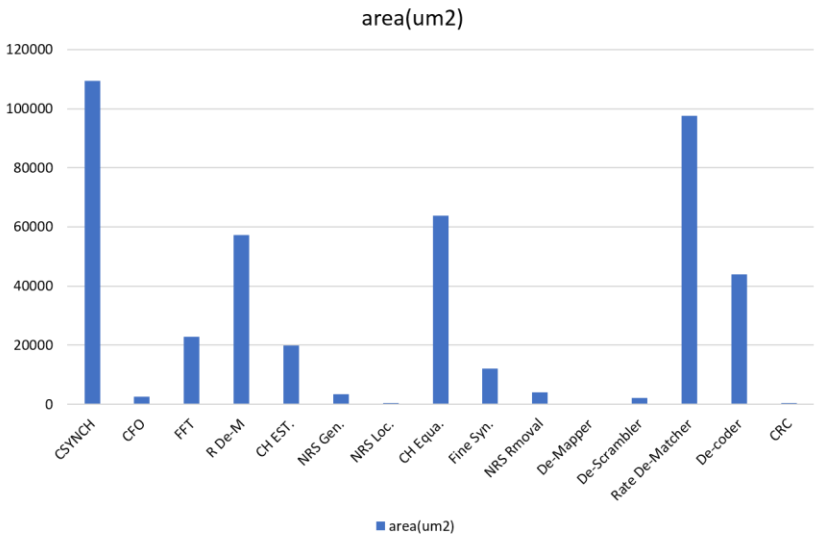


Figure 228: Area report for each block

Leakage Power (uw)	
CSYNCH	445.184
CFO	11.3075
FFT	99.34
R De-Mapper	193.65
CH EST	78.82
NRS Generation	15.3137
NRS Location	2.488
CH Equalizer	259.0588
Fine synchronization	53.1423
NRS Removal	15.93
De-mapper	111.1573
De-scrambler	9.424
Rate De-Matcher	414.2467
Channel Decoder	211.1561
CRC	2.07

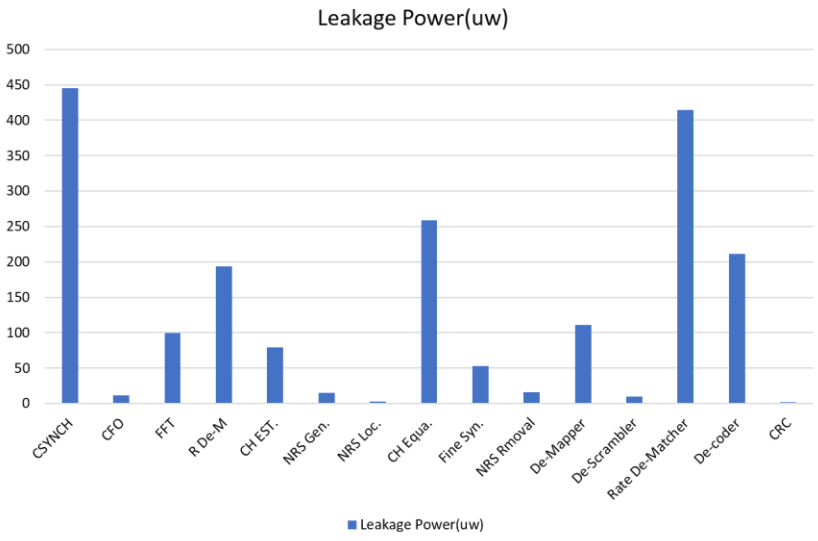


Figure 229: Power report for each block

Latency (Clock Cycle)	
Down Sampler CSYNC	1
CFO	8
FFT	19
R De-Mapper	1791
CH EST	6
NRS Generation	1606
NRS Location Max	130
NRS Location Min	2
CH Equalizer	0
Fine synchronization	15
NRS Removal	1
De-mapper	1
De-scrambler	0.5

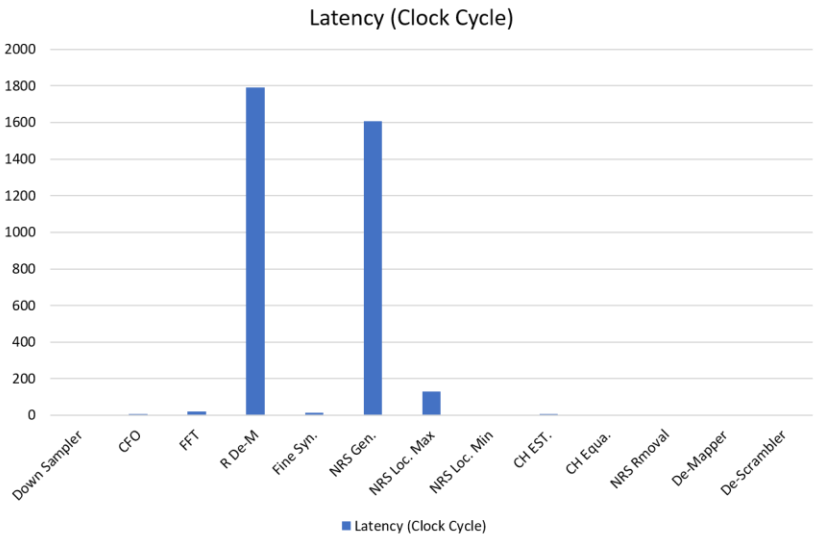


Figure 230: Latency for each block

As shown in Figure 231, Figure 232, and Figure 233 the results from DC taking into consideration that all RAMs assigned as a black box

```
Cell                    Reference       Library             Area  Attributes
-------------------------------------------------------------------------------
B1                      CFO2FFT_memory                  5201.630104
                                                                  h, n
CU                      FFT_controlunit                 23048.634422
                                                                  h, n
DUT                     Core_EqToDesc                   70152.180241
                                                                  h, n
Decoder                 top                             40245.268358
                                                                  b, h, n
Fine1                   FineSynch                       35312.830381
                                                                  h, n
Gen2                    top_NRS_GEN                     4166.624044
                                                                  h, n
SYNCRATE                mem_sharing                     210289.491530
                                                                  b, h, n
U2                      CFO_DS                          2525.138025
                                                                  h, n
U4                      BUF_X1          NangateOpenCellLibrary_ss0p95vn40c
                                                        0.798000
U5                      BUF_X1          NangateOpenCellLibrary_ss0p95vn40c
                                                        0.798000
U6                      BUF_X1          NangateOpenCellLibrary_ss0p95vn40c
                                                        0.798000
channe2                 Top_Mod_CH_EST                  19912.760110
                                                                  h, n
clkDiv                  Clock_divider                   143.906003
                                                                  h, n
crc1                    crc                             462.840006
                                                                  h, n
mult_250                integration_i_DW02_mult_0       1657.445999
                                                                  BO, h
rem                     RESOURCE_DEMAPPER               57703.911439
                                                                  h, n
rs                      FFT_storageElement              45606.499065
                                                                  h, n
top2                    top_NRS_LOC                     560.462009
                                                                  h, n
-------------------------------------------------------------------------------
Total 18 cells                                          516992.015735
```

Figure 231 Whole Chain DC Area Report

```
  Cell Internal Power  = 700.3728 uW   (65%)
  Net Switching Power  = 378.7413 uW   (35%)
                         ---------
Total Dynamic Power    =   1.0791 mW  (100%)

Cell Leakage Power     =   2.1515 mW


                 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           8.5459              0.0000             8.5459  (   0.26%)
clock_network      0.0000           0.0000              0.0000             0.0000  (   0.00%)
register           4.4542           0.5511            335.0150             5.3403  (   0.17%)
sequential       477.4288           1.7246         5.4230e+05          1.0214e+03  (  31.62%)
combinational    218.5257         367.9150         1.6087e+06          2.1951e+03  (  67.95%)
--------------------------------------------------------------------------------------------------
Total            700.4088 uW      378.7366 uW     2.1513e+06 nW       3.2304e+03 uW
```

Figure 232 Whole Chain DC Power Report

```
Des/Clust/Port     Wire Load Model       Library
------------------------------------------------
integration_i      5K_hvratio_1_1        NangateOpenCellLibrary_ss0p95vn40c

Point                                                   Incr       Path
-----------------------------------------------------------------------
clock i_clk (rise edge)                                 0.00       0.00
clock network delay (ideal)                             0.00       0.00
clkDiv/counter_4_reg[0]/CK (SDFFR_X1)                   0.00       0.00 r
clkDiv/counter_4_reg[0]/Q (SDFFR_X1)                    0.08       0.08 r
clkDiv/add_34/A[0] (Clock_divider_DW01_inc_1)           0.00       0.08 r
clkDiv/add_34/U1_1_1/CO (HA_X1)                         0.06       0.14 r
clkDiv/add_34/U1_1_2/CO (HA_X1)                         0.06       0.20 r
clkDiv/add_34/U1_1_3/CO (HA_X1)                         0.06       0.26 r
clkDiv/add_34/U1_1_4/CO (HA_X1)                         0.06       0.31 r
clkDiv/add_34/U1_1_5/CO (HA_X1)                         0.06       0.37 r
clkDiv/add_34/U2/Z (XOR2_X1)                            0.06       0.43 r
clkDiv/add_34/SUM[6] (Clock_divider_DW01_inc_1)         0.00       0.43 r
clkDiv/counter_4_reg[6]/SE (SDFFR_X1)                   0.01       0.44 r
data arrival time                                                  0.44

clock i_clk (rise edge)                                20.00      20.00
clock network delay (ideal)                             0.00      20.00
clock uncertainty                                      -0.35      19.65
clkDiv/counter_4_reg[6]/CK (SDFFR_X1)                   0.00      19.65 r
library setup time                                     -0.08      19.57
data required time                                                19.57
-----------------------------------------------------------------------
data required time                                                19.57
data arrival time                                                 -0.44
-----------------------------------------------------------------------
slack (MET)                                                       19.13
```

Figure 233 Whole Chain DC Timing Report

4.3.2. VIVADO results:

Resource	Utilization	Available	Utilization %
LUT	50728	433200	11.71
LUTRAM	3016	174200	1.73
FF	31896	866400	3.68
BRAM	69	1470	4.69
DSP	80	3600	2.22
IO	4	850	0.47

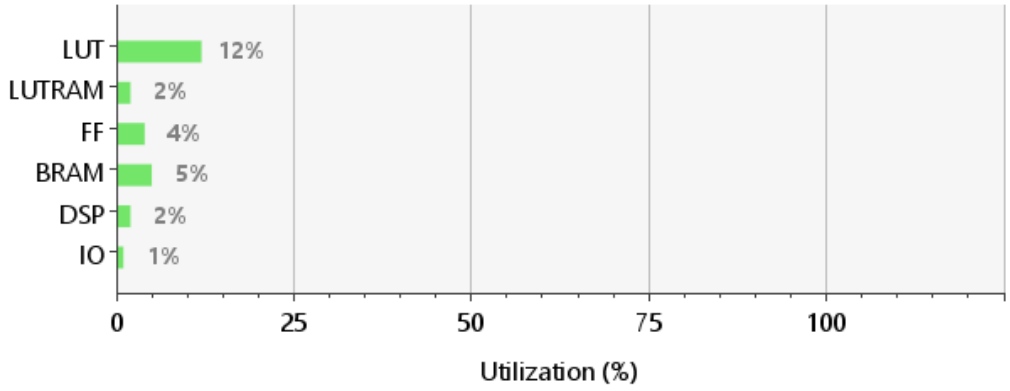


Figure 234 VIVADO Utilization for Vertex 7 for Whole Chain

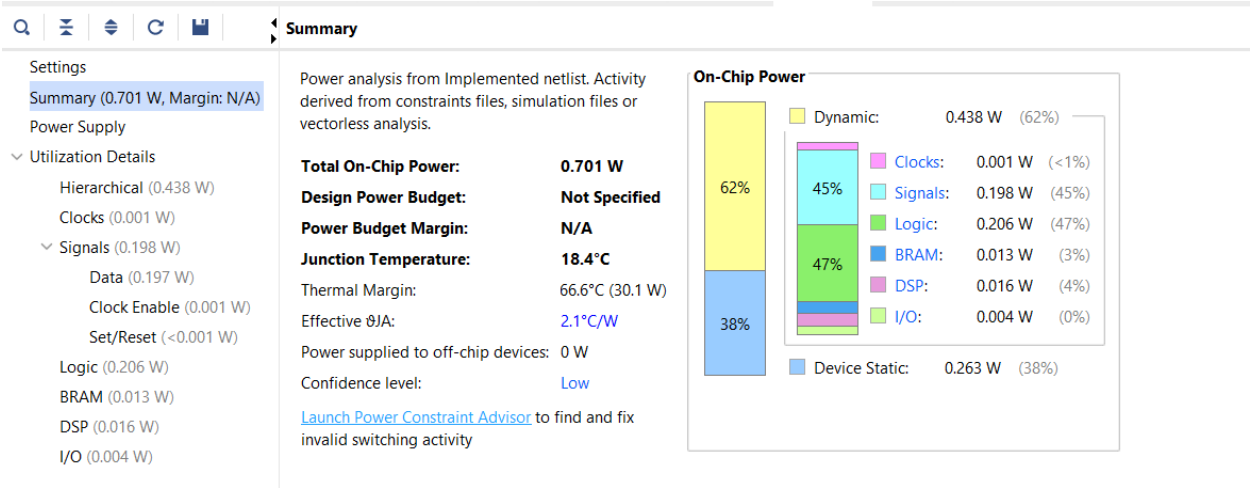


Figure 235 VIVADO power summary for Vertex 7 for Whole Chain

4.4.FPGA implementation:

The generated files from MATLAB are stored into a ROM and make the interface of the chain to the ROM. By using the debugging tools in VIVADO, Virtual Input Output (VIO) and Integrated Logic Analyzer (ILA), the output should be observed from the chain to make sure that the results are correct. Also, we used the MCMM in FPGA to generate deferent clock domains to make sure that the phases between them does not change as time passes.

As shown in Figure 237, the outputs from ILA FPGA: nrsRemoval indicates to the output from P/S & NRS removal block, demod_concat which indicated to the output from demodulation block, desc_concat which indicates to the output from descrambler block and out_concat which indicates to the final output from the chain which all of them are in unsigned radix and they are equal to the results from behavioral simulation in Figure 236.

Also, the VIO in FPGA we have two push buttons, one for reset button and one for enable button for the system. If the enable button is 1 and reset button is 1 –negative edge reset- the output from the FPGA is as same as in simulation, we take a sample from each block to make sure that the blocks work correctly, and the Final Ack signal indicates that the data is correct as shown in Figure 238.

If the reset signal is 0 and the enable signal 0 or 1 there is no output from the FPGA as shown in Figure 239. Also, if the enable signal is 0 and reset signal is 1 there is no output from the chain as shown in Figure 240.

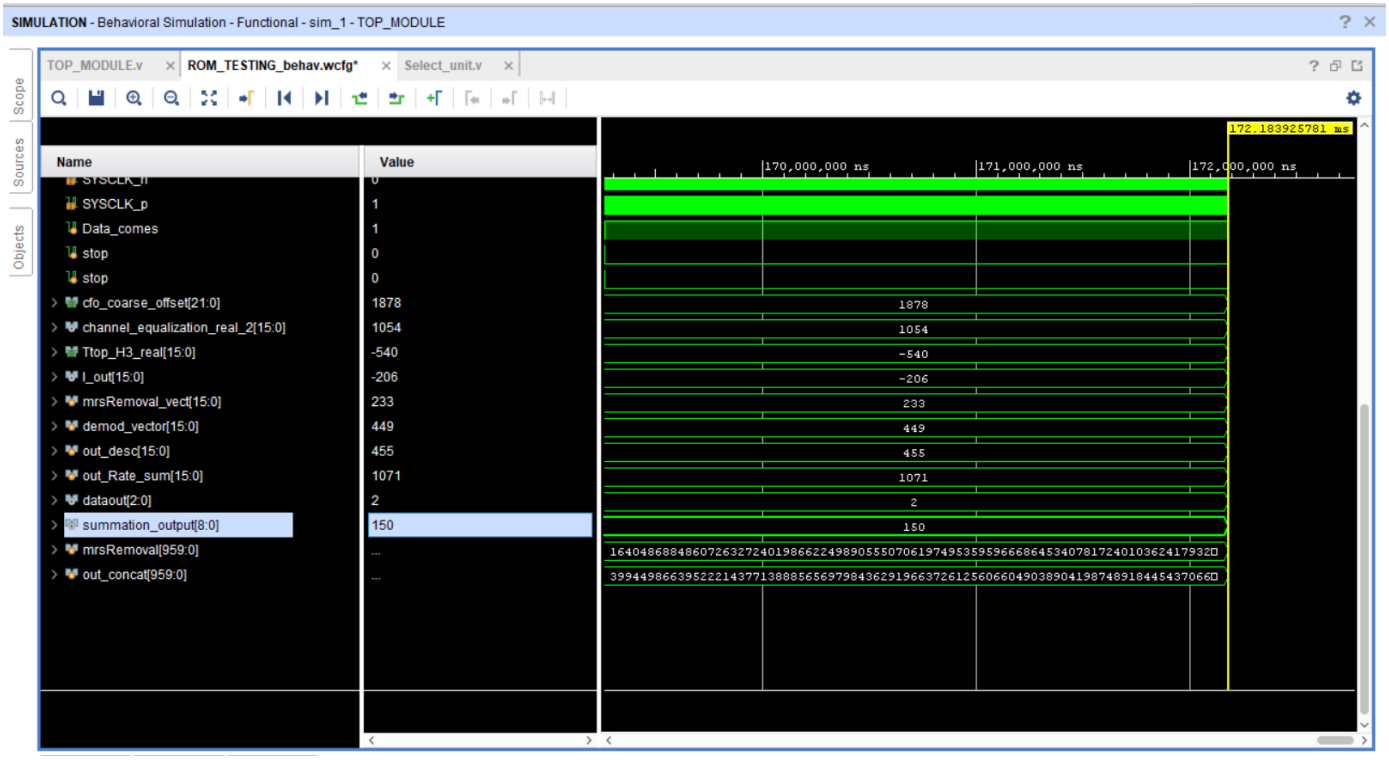


Figure 236 VIVADO Behavioral Simulation

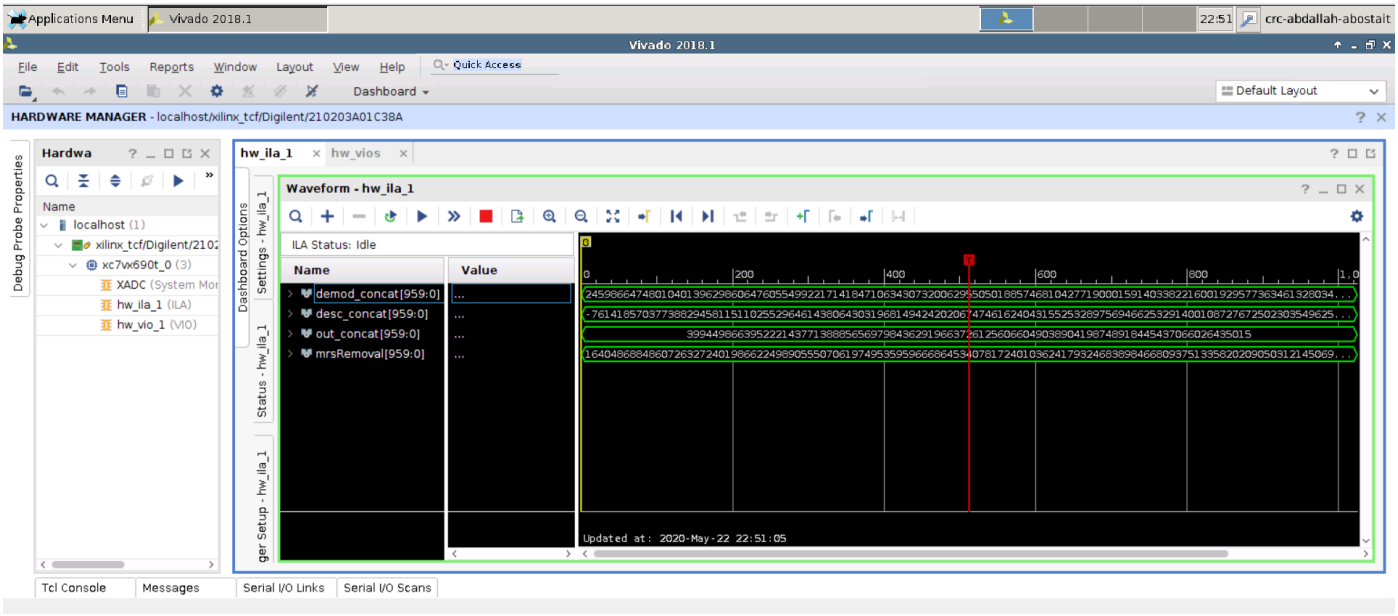


Figure 237 ILA FPGA Output

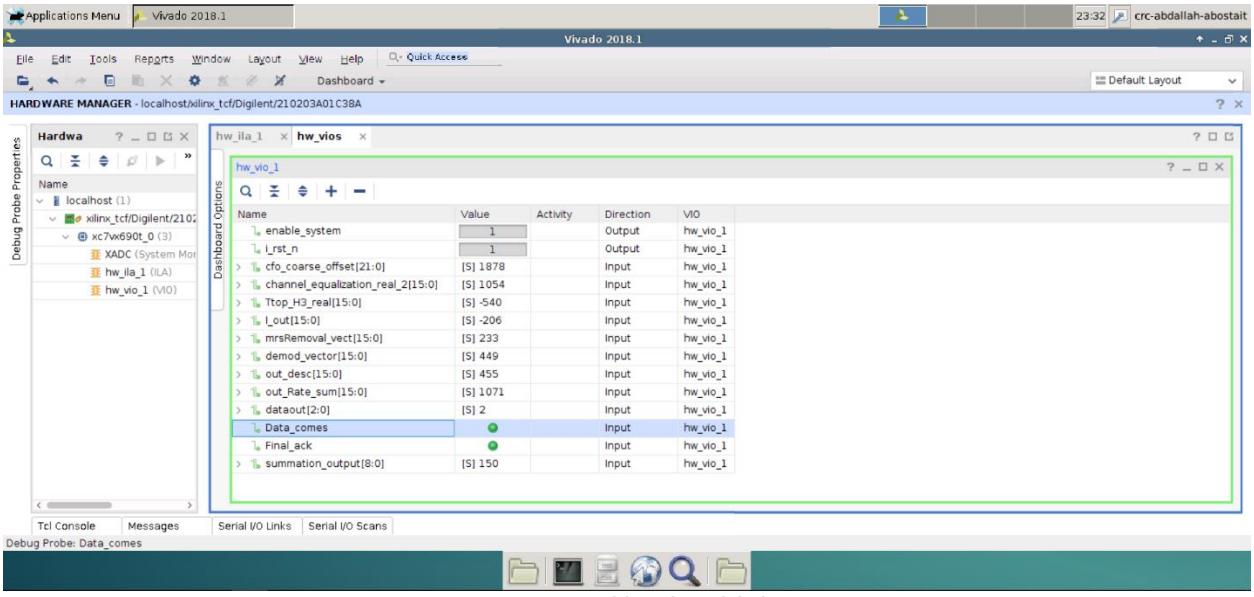


Figure 238 VIO FPGA Output

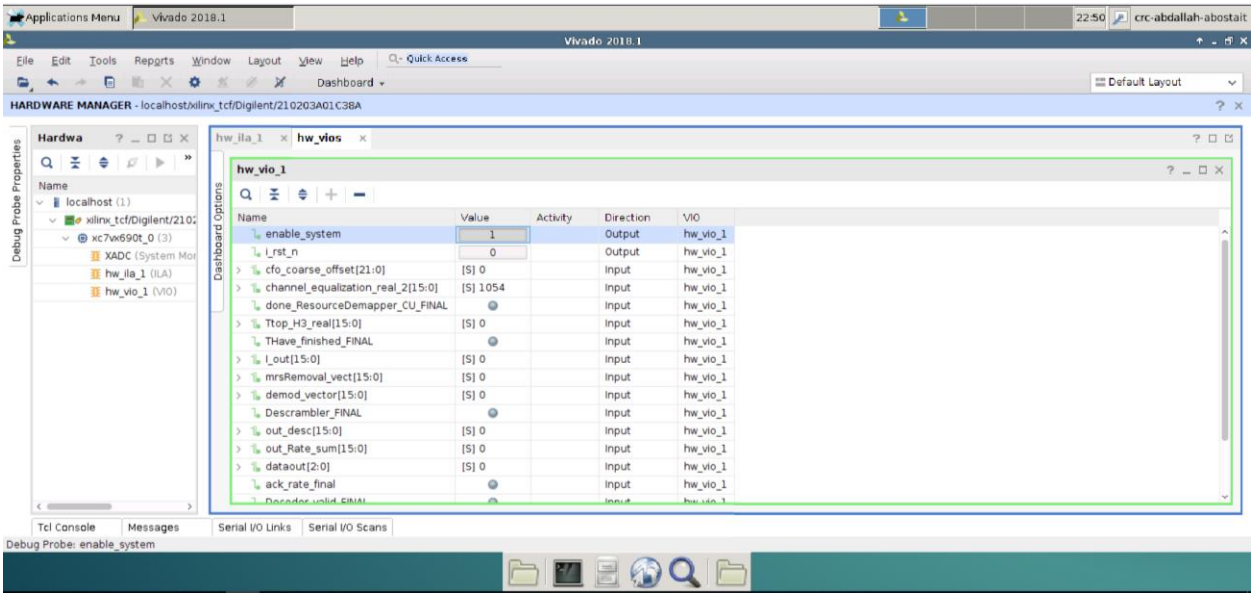


Figure 239 FPGA Output if reset is 0

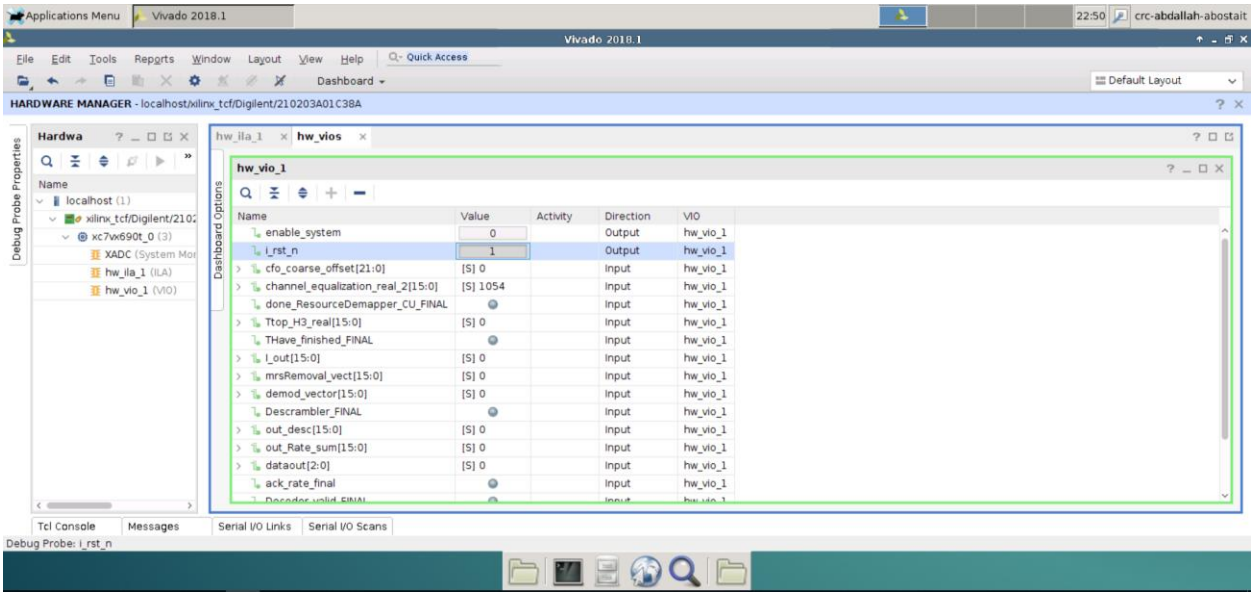


Figure 240 FPGA Output if enable 0

Chapter 5
Conclusion and Future Work

In this thesis NPDSCH chain implementation have presented. Walked through algorithm design, hardware implementation, MATLAB and RTL simulations and synthesis for each block in the chain, then after chain integration, MATLAB and RTL chain simulations, chain synthesis and testing on FPGA Vertex7. This chain has a maximum rate of 98.666Kbps and a latency of 1.45ms. The chain meets all 3GPP standard specifications. Our future work is sharing more memory resources across the chain, implementing soft de-modulation and soft decoding to enhance performance, and optimizing on block level to reduce area and power.

References

[1] Narrow Band LTE- NB-IoT – Codeplayon, Codeplayon.

[2] NB-IoT Applications in Smart Grid: Survey and Research Challenges

[3] Evolution of NB-IoT and its implementation, simnovus.

[4] Enhancements of Narrowband IoT in 3GPP Rel-14 and Rel-15 Rapeepat Ratasuk, Nitin Mangalvedhe, Zhilan Xiong, Michel Robert, David Bhatoolaul Mobile Radio Research Lab, Nokia Bell Labs.

[5] 3GPP TS_136211 V14.11.0 (2019-06).

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

[7] Qualcomm Incorporated, "NB-PSS and NB-SSS Design," 3GPP TSG RAN WG1 Ad-Hoc Meeting, Tech. Rep. R1-161981, March 2016.

[8] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Base Station (BS) radio transmission and reception (Release 14) Ver. 14.9.0, document TS 36.104, Mar. 2019.

[9] 3rd Generation Partnership Project Technical Specification Group Radio Access Network Evolved Universal Terrestrial Radio Access (E-UTRA) Physical channels and modulation (Release 14) Ver. 14.9.0, document TS 36.211, Mar. 2019.

[10] W. Yang et al., "Enhanced System Acquisition for NB-IoT," in IEEE Access, vol. 5, pp. 13179-13191, 2017, doi: 10.1109/ACCESS.2017.2724601.

[11] 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.

[12] H. Kroll, M. Korb, B. Weber, S. Willi, and Q. Huang, "Maximum likelihood detection for energy-efficient timing acquisition in NB-IoT," in Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW), Mar. 2017, pp. 15.

[13] Praween Kumar Nishad, P.Singh, "Carrier Frequency Offset Estimation in OFDM Systems", IEEE,2013.

[14] N. Prasad, Ayas Swain, Kamalakanta Mahapatra, "FPGA Implementation of Low Latency Scaled CORDIC Based Discrete Fourier Transform Core", ICEEE,2012.

[15] K.Vinoth Babu and 2 Dr. G. Ramachandra Reddy 3 Sunit Gupta, 4Utpal Sharma, 5 Patel Pratik "STUDY AND ANALYSIS OF VARIOUS FREQUENCY OFFSET CORRECTION SCHEMES IN ORTHOGONAL FREQUENCY DIVISION MULTIPLEXING (OFDM) BASED LONG TERM EVALUATION (LTE) SYSTEMS"

[16] A COMPARATIVE ANALYSIS OF FFT ALGORITHMS

[17] Fast Fourier Transform Architectures: A Survey and State of the Art, Anwar Bhasha Pattan, Dr. M. Madhavi Latha, Research Scholar, Dept. of ECE, JNTUH, Hyderabad, India, Professor, Dept. of ECE, JNTUH, Hyderabad, India

[18] Implementation of Fast Fourier Transform (FFT) on FPGA using Verilog HDL

[19] FPGA implementation of Radix-22 Pipelined FFT Processor AHMED SAEED1, M. ELBABLY2, G. ABDELFADEEL2 and M. I. ELADAWY2 1Electrical Engineering, 2Electronics & Communication Engineering 1Future University in Egypt, 2Helwan University Helwan City, Cairo EGYPT

[20] Pipelined Radix-2 Feedforward FFT Architectures Mario Garrido, Member, IEEE, J. Grajal, M. A. Sánchez, and Osear Gustafsson, SéniorMember, IEEE.

[21] An area-efficient and low-power 64-point pipeline Fast Fourier Transform for OFDM applications.

[22] A New Approach to Pipeline FFT Processor Shousheng He and Mats Torkelson Department of Applied Electronics, Lund University, S-22100 Lund, SWEDEN.
[23] Parallel Extensions to Single-Path Delay-Feedback FFT Architectures Brett W. Dickson, and Albert A. Conti.
[24] Design of Combined Radix-2, Radix- 4 and Radix-8 based Single Path Delay Feedback (SDF) FFT.
[25] Design of 16-point Radix-4 Fast Fourier Transform in 0.18μm CMOS Technology 1Siva Kumar Palaniappan and 2Tun Zainal Azni Zulkifli 1RFIC Design Group, Faculty of Electrical and Electronics Engineering, Universiti Sains Malaysia 2Engineering Campus, 14300 Nibong Tebal, Seberang Perai Selatan, Penang, Malaysia.
[26] Implementing the Radix-4 Decimation in Frequency (DIF) Fast Fourier Transform (FFT) Algorithm Using a TMS320C80 DSP.
[27] 64 Point Radix-4 FFT Butterfly Realization using FPGA Amaresh Kumar, U.N. Tripathi, Roopak Kumar Verma, Manish Mishra Department of Electronics, D.D.U. Gorakhpur University, Gorakhpur Department of Computer Science, D.D.U. Gorakhpur University, Gorakhpur.
[28] Split-Radix Fast Fourier Transform Using Streaming SIMD Extensions Version 2.1
[29] Split-Radix FFT Algorithms Based on Ternary Tree Ming Zhang School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China.
[30] Low-Power Split-Radix FFT Processors Using Radix-2 Butterfly Units Zhuo Qian and Martin Margala.
[31] An Area Efficient 1024-point Low Power Radix-22 FFT Processor with Feed-forward Multiple Delay Commutators Ngoc Le Ba, Student Member, IEEE, and Tony Tae-Hyoung Kim, Senior Member, IEEE.
[32] DESIGN AND IMPLEMENTATION OF TRANSMITTER CHAIN FOR MACHINE TYPE COMMUNICATION ON LTE NETWORKS Submitted by - Hemanth Bollamreddi Department of Electrical Engineering Indian Institute of Technology, Kanpur Email: blmhemu@iitk.ac.in Supervised by - Dr. Rohit Budhiraja Department of Electrical Engineering Indian Institute of Technology, Kanpur Email: rohitbr@iitk.ac.in
[33] 3GPP TS 36.211: "Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Channels and Modulation".
[34] 3GPP TS 36.104: "Evolved Universal Terrestrial Radio Access (E-UTRA); Base Station (BS) radio transmission and reception".
[35] Douglas Torha and Piotr Krasowski, Wireless system design NB-IoT downlink simulator, 2017.
[36] Asad Mahmood and Waqas Aslam Cheema, Channel Estimation for LTE downlink, 2009.
[37] 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.
[38] Yong Soo Cho, Jaekwon Kim, Won Young Yang and Chung G. Kang, MIMO-OFDM Wireless Communications with MATLAB, 2010.
[39] 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).
[40] A Low-Power Implementation of arctangent function for Communication Applications using FPGA M. saber, Yutaka Jitsumatsu and T. Kohda.
[41] Analysis of Circular Buffer Rate Matching for LTE Turbo Code Jung-Fu (Thomas) Cheng*, Ajit Nimbalker+, Yufei Blankenship+, Brian Classon+, and T. Keith Blankenship+* Ericsson Research, RTP, NC, USA.

[42] An Improved Rate Matching Algorithm for 3GPP LTE Turbo Code Long Yu1, Xu Wang1, Jian Liu2,1.
[43] Reduced Complexity Rate-matching/De-matching Architecture for the LTE Turbo Code Angelos Spanos*, Fotios Gioulekas†, Michael Birbas‡, Athanasios Vgenis‡ *The University of Edinburgh, Scotland, UK.
[44] Automated performance-based design technique for an efficient LTE PDSCH implementation using.
[45] SDSoC tool Mohamed Eladawy1 | Mahmoud Mostafa1 | M. Sameh Said1 | Hassan Mostafa1,2.
[46] Research of Communication Rate Matching Algorithm for LTE Physical Layer 1Nina Dai 1Signal and Information Processing Key Lab, Chongqing Three Gorges University, Chongqing 404000, China.
[47] Optimized Rate Matching Architecture for a LTEAdvanced FPGA-based PHY Karlo G. Lenzi, Jose A. Bianco F., Felipe A. de Figueiredo, Fabricio L. Figueiredo.
[48] Analisi di protocolli HARQ per il sistema cellulare LTE.
[49] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and channel coding (Release 14) Ver. 14.9.0, document TS 36.212, Mar. 2019.
[50] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Layer Procedures (Release 14) Ver. 14.9.0, document TS 36.213, Mar. 2019.
[51] Lee, Jungwon, et al. "Effect of carrier frequency offset on OFDM systems for multipath fading channels." IEEE Global Telecommunications Conference, 2004. GLOBECOM'04.. Vol. 6. IEEE, 2004.