NAME : ARNAB KUMAR MONDAL :

UIN : 326006709.

LDPC DECODER REPORT :

LDPC is an effective Error Correcting code for communication and storage channels. In the project we use a parallel LDPC decoder which is represented by a tanner graph which looks like this :

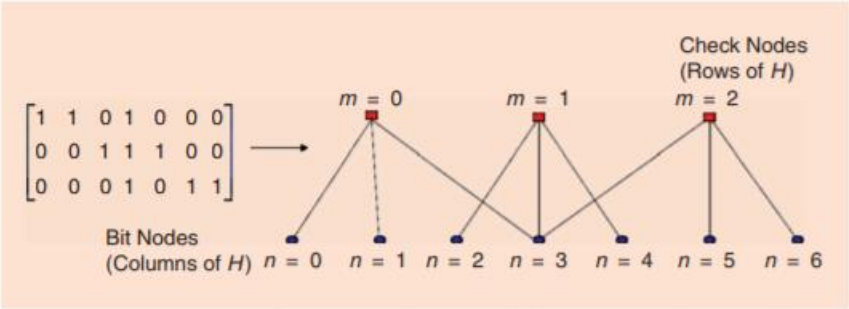


FIGURE 2: Representations of an LDPC code: parity-check matrix *H* and its equivalent Tanner graph.

The parity check matrix H which is given as an input to the LDPC decoder consists of the relation between the check node units (CNU) and the variable node units (VNU). Wherever CNU and Vnu are related they are set as 1 and the tanner graph is created in this fashion.

The files that were provide to us were :

1. script_LDPC is the main file
2. InitializeWimaxLDPC.m is a function which gets called in main file
3. H_matrices_802_16e.mat contains data.
   H_matrices_802_16e.mat is a .mat file which contains H matrix(parity check matrix). This H matrix is in exponent form. The function InitializeWimaxLDPC.m converts exponent form to binary form. This function is called in script_LDPC.m @ line number 82. So at line number 82 we get the parity check matrix in binary form. We can see that it is stored in variable named Henc. We had to write fixed point and floating point ldpc parallel decoder in MATLAB for this particular H matrix only.
   script_LDPC.m will perform some simulations on the ldpc parallel decoder and print bit error rate and frame error rate statistics for different snr.

In this project for the variable node implementation I used the following file in MATLAB and we use soft decision decoding :

1. ldpc_parallel.m : In this MATLAB file the parity check matrix H and the llr is given as an input. At first the tanner graph was created as g1 which included the connections between the various CNU and VNU. The tanner graph g1 consists of 1824 edges. The sign of the llr value was assigned to the array llr_sign.
   llr_sign[i] = 0 llr(vnu_no)>0
   = 1 llr(vnu_no) <0.

The syndrome was calculated as XOR(H,llr_sign) and it determines whether the data is correct or not based on the number of non zeros in the syndrome array.

Syndrome=mod((H*llr),2)

If the number of non zeros in the syndrome is not equal to zero then we performed the correction operation. The sign of each edge was calculated as XOR(syndrome(cnu),llr_sign(vnu)) and was denoted as :

$$\text{sign}(R_{m\to n}^{(l)}) = \text{sign}(R_{n\to m}^{(l-1)}) \oplus s_m.$$

For finding the |R| of any edge we have to evaluate the absolute minimum Q value of its neighbouring nodes. Q_{VNU_no}=llr(vnu_no). Now based on the sign of each edge |R|=R or |R|=-R. After that we take the sum of R of edges corresponding to a particular vnu and define it as llr2. Make a copy of llr as llr1 and then take llr1=llr1+llr2. After this check the sign of llr1 array and find the syndrome again. If the syndrome is zero we break away from the operation. If not we continue till the number of non zero elements in the syndrome is 0. Sometimes the operation may have an infinite loop and the graph does not hold that value. So we cap the iterations till 200. The loop works only when the count is less than 200 and the number of non zeros in the syndrome is not equal to zero. This LDPC decoder is designed for the floating point LDPC decoder and is therefore not practical.

2. Quantize.m : This gives us the quantizer where we quantize the LLR values so that the LDPC works for the fixed order which can be realized in hardware.
3. ldpc_parallel_quantized.m : This is a fixed point LDPC decoder which works in the same way as that of the floating point LDPC decoder except that the llr's in this case are quantized by the Quantize function. llr1 and llr are quantized by the quantize function.

For the Verilog operation of the fixed point algorithm we used 3 files for the design and 3 files for the testbench.

The design files are as follows:

1. Ldpc_Decoder_final.v : This file gives us the connection between the CNU and VNU. This is found out from the graph in Matlab and the code was printed using 2 files read1.c and prog1.c . The CNU is connected to the VNU and the VNU is connected to the CNU. Some of the CNU have 6 edges and some have 7 edges and thus they are denoted as CNU_6 and CNU_7 respectively. Similarly the VNU's have 2,3 and 6 CNU's connected to it and thus they are defined in the modules VNU_2,VNU_3 and VNU_6 respectively. It takes in the llr values as input and gives a bit sequence P of length 576.
2. CNU_6 and CNU_7 : These are for the CNU description purpose. These take Q (VNU LLR's) as input and gives us R which is |R| as output.
3. VNU_2, VNU_3 and VNU_6 : These are for the VNU description purpose. They take the R values (described above) and L (current LLR of a particular VNU) as input and give the Q values (updated LLR values) and P (output bit pattern) as outputs.
4. ldpc_Decoder_final_tb.v: This is the testbench for our design, which instantiates Ldpc_Decoder_final module and generates the output bit pattern of the ldpc decoder.
5. CNU_6_tb.v: This is the testbench for CNU_6, which instantiates CNU_6 module and generates R.
6. VNU_6_tb.v: This is the testbench for VNU_6, which instantiates VNU_6 module and generates Q and P.

OUTPUTS :

FOR QUANTIZED OPERATION :

```
SNR is 1: ber: 0.154844; fer: 1.000000
SNR is 1.400000e+00: ber: 0.121342; fer: 0.970874
SNR is 1.800000e+00: ber: 0.105089; fer: 0.900901
SNR is 2.200000e+00: ber: 0.064515; fer: 0.729927
SNR is 2.600000e+00: ber: 0.031132; fer: 0.423729
SNR is 3: ber: 0.016641; fer: 0.278552
```

FOR NON-QUANTIZED OPERATIONS

```
SNR is 1: ber: 0.145283; fer: 0.990099
SNR is 1.400000e+00: ber: 0.122162; fer: 0.961538
SNR is 1.800000e+00: ber: 0.095584; fer: 0.943396
SNR is 2.200000e+00: ber: 0.054533; fer: 0.684932
SNR is 2.600000e+00: ber: 0.034021; fer: 0.505051
SNR is 3: ber: 0.014662; fer: 0.274725
SNR is 3.400000e+00: ber: 0.004769; fer: 0.102145
```

Quantization Degradation :

SNR	Quantized BER.	Non Quantized BER.	Quantization Degradation in BER	Quantized FER.	Non-Quantized FER.	Quantization Degradation in BER
1	0.154844	0.145283	9.561E(-3)	1	0.990099	9.9E(-3)
1.4	0.121342	0.122162	-8.2E(-4)	0.970874	0.961538	9.336E(-3)
1.8	0.105089	0.095584	9.505E(-3)	0.900901	0.943396	-4.2E(-2)
2.2	0.064515	0.054533	9.982E(-3)	0.729927	0.684932	4.499E(-2)
2.6	0.031132	0.034021	-2.889E(-3)	0.423729	0.505051	-8.13E(-2)
3.0	0.016641	0.014662	1.979E(-3)	0.278552	0.274725	3.827E(-3)

The outputs for non quantized and quantized are not going above 3 and 3.4 because the BER and FER goes to zero and thus the output is not printed because the output is printed only when BER and FER is greater than 0.

VERILOG OUTPUT FOR THE LDPC TEST BENCH :

VNU_6_tb.v

```
ModelSim> vsim -gui work.CNU_6_tb
# vsim
# Start time: 22:46:27 on Apr 30,2018
# Loading work.CNU_6_tb
# Loading work.CNU_6
VSIM 9> run -all
# Inputs;           x :           x :           x :           x :           x :           x
# Outputs;          x :           x :           x :           x :           x :           x
# Inputs;          -2 :          -4 :           5 :          -6 :           7 :           8
# Outputs;          x :           x :           x :           x :           x :           x
# Inputs;          -2 :          -4 :           5 :          -6 :           7 :           8
# Outputs;          4 :           2 :          -2 :           2 :          -2 :          -2
# Inputs;          -2 :          -4 :           5 :          -6 :           7 :           8
# Outputs;          4 :           2 :          -2 :           2 :          -2 :          -2
# Inputs;          -2 :          -4 :           5 :          -6 :           7 :           8
# Outputs;          4 :           2 :          -2 :           2 :          -2 :          -2
# ** Note: $finish    : C:/Users/ARNAB KUMAR MONDAL/Desktop/CNU_6_tb.v(41)
#    Time: 170 ns  Iteration: 0  Instance: /CNU_6_tb
# 1
```

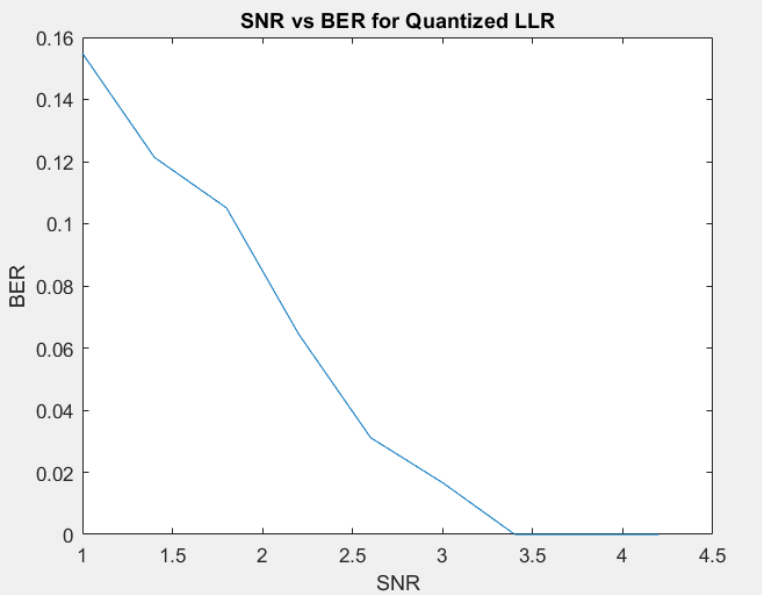
CNU_6_tb.v

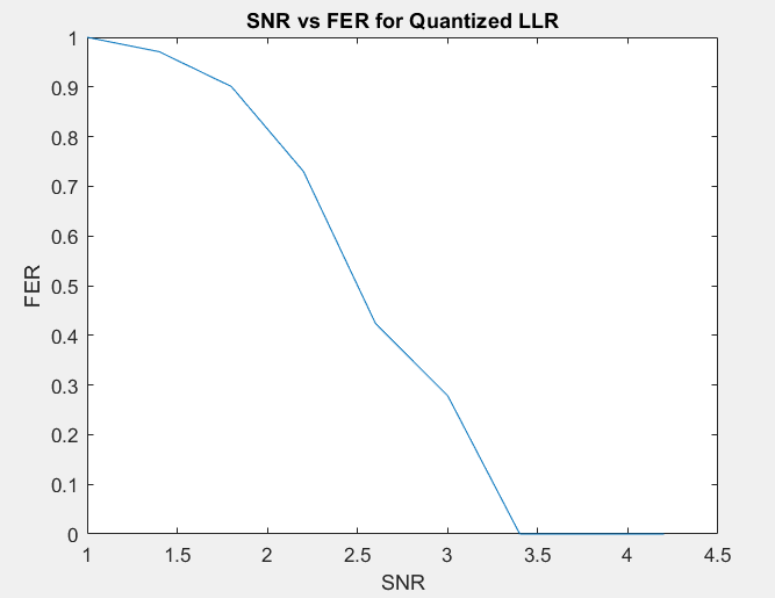
```
# Start time: 22:47:56 on Apr 30,2018
# Loading work.VNU_6_tb
# Loading work.VNU_6
VSIM 12> run -all
# Output Q values :          2 :          2 :          2 :          2 :          2 :          2 ; Output decoded value : 0
# Input R values ;          x :          x :          x :          x :          x :          x  ; Input L value :            2
# Output Q values :          2 :          2 :          2 :          2 :          2 :          2 ; Output decoded value : 0
# Input R values ;         -4 :         10 :          5 :         -6 :          7 :          8  ; Input L value :            4
# Output Q values :         28 :         14 :         19 :         30 :         17 :         16 ; Output decoded value : 0
# Input R values ;         -4 :          1 :        -11 :         -6 :          7 :          6  ; Input L value :            4
# Output Q values :          1 :         -4 :          8 :          3 :        -10 :         -9 ; Output decoded value : 1
# Input R values ;         -4 :          1 :        -11 :         -6 :          7 :          6  ; Input L value :            4
# Output Q values :          1 :         -4 :          8 :          3 :        -10 :         -9 ; Output decoded value : 1
# Input R values ;         -4 :          1 :        -11 :         -6 :          7 :          6  ; Input L value :            4
# ** Note: $finish    : C:/Users/ARNAB KUMAR MONDAL/Desktop/VNU_6_tb.v(58)
#    Time: 190 ns  Iteration: 0  Instance: /VNU_6_tb
```

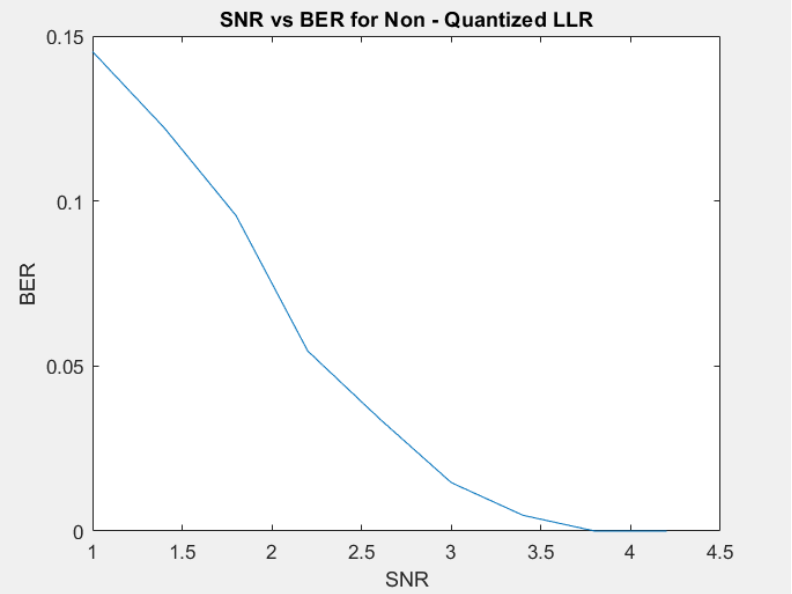
Ldpc_Decoder_final_tb.v

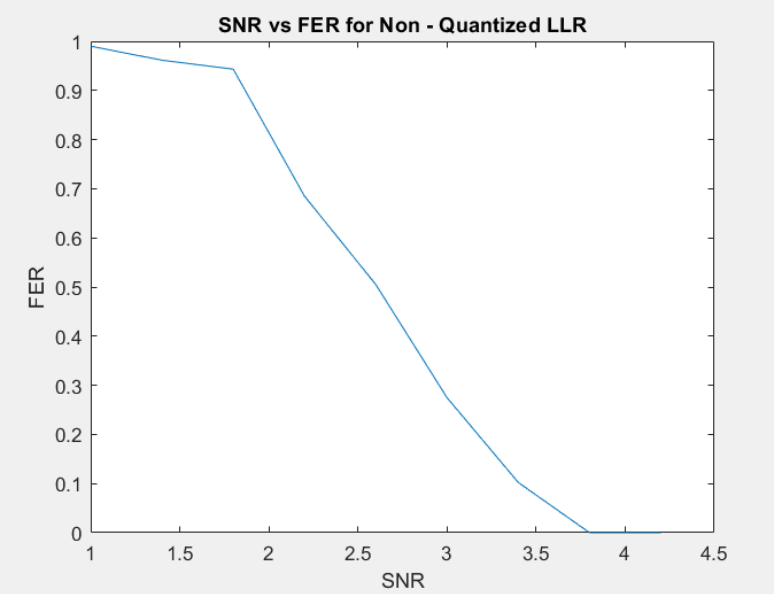
```
# Loading work.VNU_6
# Loading work.VNU_2
VSIM 4> run -all
# xxxxxxxxxxxx002020xxxxxxxxxxxx430200xxxxxx402001xxxxxx000a00xxxxxx080800xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# xxxxxxxxxxxx002020xxxxxxxxxxxx430200xxxxxx402001xxxxxx000a00xxxxxx080800xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# xxxxxxxxxxxx002020xxxxxxxxxxxx430200xxxxxx402001xxxxxx000a00xxxxxx080800xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# xxxxxxxxxxxx002020xxxxxxxxxxxx430200xxxxxx402001xxxxxx000a00xxxxxx080800xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# xxxxxxxxxxxx002020xxxxxxxxxxxx430200xxxxxx402001xxxxxx000a00xxxxxx080800xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# xxxxxxxxxxxx002020xxxxxxxxxxxx430200xxxxxx402001xxxxxx000a00xxxxxx080800xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# xxxxxxxxxxxx002020xxxxxxxxxxxx430200xxxxxx402001xxxxxx000a00xxxxxx080800xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# xxxxxxxxxxxx002020xxxxxxxxxxxx430200xxxxxx402001xxxxxx000a00xxxxxx080800xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# xxxxxxxxxxxx002020xxxxxxxxxxxx430200xxxxxx402001xxxxxx000a00xxxxxx080800xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# 4020900000110000200c0012004304441420010113000000400200a0009000000100008002002040840a06049104004232000002482200206008011920080202001002000000421040
# 4020900000110000200c0012004304441420010113000000400200a0009000000100008002002040840a06049104004232000002482200206008011920080202001002000000421040
# 40a28292002000240044008200218242020008112040040420200a880800a80000082008208000004080604810610433004000002200120202801190011120200100a00000421840
# 40a28292002000240044008200218242020008112040040420200a880800a80000082008208000004080604810610433004000002200120202801190011120200100a00000421840
# 400290300118000400 0c0082400282020a1008112042004560000a0808204800100 8212d2000011044a0604020600023200000 20e00018021040109200002000010000000 80420000
# 400290300118000400 0c0082400282020a1008112042004560000a0808204800100 8212d2000011044a0604020600023200000 20e00018021040109200002000010000000 80420000
# 500290810039020000 0c0082400300034a10001100400004202002890b006800000428302082050844b06049104000330800002020041 01212c41190810028200500000044421040
# 500290810039020000 0c0082400300034a10001100400004202002890b006800000428302082050844b06049104000330800002020041 01212c41190810028200500000044421040
# 88061281203b0004044c0012004300404a02001120001404602002808820400020082908a080040840a160490040003308c00004a200101202c50182000820a00582e00084061840
# 88061281203b0004044c0012004300404a02001120001404602002808820400020082908a080040840a160490040003308c00004a200101202c50182000820a00582e00084061840
# 04428080001000242 00c008200420142028200111040 01c040000a088b00801000002028208015 0044a020410041003a200000008a048212008611921190202004006 00000000840
# 04428080001000242 00c008200420142028200111040 01c040000a088b00801000002028208015 0044a020410041003a200000008a048212008611921190202004006 00000000840
# 40009083001000243 44c029240010e41020008113002240000000 2080b00a000100000002 0c0010040a060410060003320000024020002 0202c45192011000a00100e04000421048
# 40009083001000243 44c029240010e41020008113002240000000 2080b00a000100000002 0c0010040a060410060003320000024020002 0202c45192011000a00100e04000421048
# 4000800300070004044400120003044002010011006000440 0200a080b20600000000200020000410 44a04049124000332080002 0aa00101200c1002201002020018 0a00080460040
# 4000800300070004044400120003044002010011006000440 0200a080b20600000000200020000410 44a04049124000332080002 0aa00101200c1002201002020018 0a00080460040
# 50a29081013000200 04c10120043070188020811404015002820 0a80ca00884038082819200201104020504910540023204000242a009206004211840111200001000040 00401008
# 50a29081013000200 04c10120043070188020811404015002820 0a80ca00884038082819200201104020504910540023204000242a009206004211840111200001000040 00401008
# 00c28280001000200 02c000240622b404800000134a0000500400a0088406000380021202000440841a060580060047b2000410420001012100405000010202004006 00004461240
# 00c28280001000200 02c000240622b404800000134a0000500400a0088406000380021202000440841a060580060047b2000410420001012100405000010202004006 00004461240
# 48869091001240202 04c0282002384434a00201114c020004010029908002000100 8212c2080040840a020491641002128000000000040012120011920000 20200c20e02000461040
# 48869091001240202 04c0282002384434a00201114c020004010029908002000100 8212c2080040840a020491641002128000000000040012120011920000 20200c20e02000461040
# 04208291221100040 08c001240c382420a102011308041 0c28104a0c08004000b0000900a00015004400204906010032200041 24a00038020080119201180020050060 0000420040
# 04208291221100040 08c001240c382420a102011308041 0c28104a0c08004000b0000900a00015004400204906010032200041 24a00038020080119201180020050060 0000420040
# 10428080002040242 10c009220030443420008 1184c004002000020809002000000001a92000050800806049126000322000002423009002126410960111202201206 00080421000
# 10428080002040242 10c009220030443420008 1184c004002000020809002000000001a92000050800806049126000322000002423009002126410960111202201206 00080421000
# 540280000011000400 4c00120001064280000011004010054010 0a000800c8c018002820a080040044a06041044100320040000422801 00200c21192010320000102004082020000
# 540280000011000400 4c00120001064280000011004010054010 0a000800c8c018002820a080040044a06041044100320040000422801 00200c21192010320000102004082020000
# 400280800012000000 4c4002100a0640080000112040004020100288090028001000001820024100458050491060003128000024820008 1200c2119e00000020010000000061400
# 400280800012000000 4c4002100a0640080000112040004020100288090028001000001820024100458050491060003128000024820008 1200c2119e00000020010000000061400
# 400090810021022004 4c00024021000082000011204004002010020808002000880060002080040044a040010040402b20000004a00080121 0c0118a020420200c00204000421000
# 400090810021022004 4c00024021000082000011204004002010020808002000880060002080040044a040010040402b20000004a00080121 0c0118a020420200c00204000421000
# ** Note: $finish    : C:/Users/ARNAB KUMAR MONDAL/Desktop/Ldpc_decoder_final_tb.v(637)
#    Time: 205 ns  Iteration: 0  Instance: /Ldpc_decoder_final_tb
```

GRAPHS :


SNR vs BER for Quantized LLR
BER
SNR
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
1
1.5
2
2.5
3
3.5
4
4.5


SNR vs FER for Quantized LLR
FER
SNR
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
1.5
2
2.5
3
3.5
4
4.5


SNR vs BER for Non - Quantized LLR
BER
SNR
0.15
0.1
0.05
0
1
1.5
2
2.5
3
3.5
4
4.5


SNR vs FER for Non - Quantized LLR
FER
SNR
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
1.5
2
2.5
3
3.5
4
4.5

OBSERVATIONS AND FUTURE SCOPE:

We observe that with increasing SNR the bit error rate decreases and the frame error rate also decreases. It is also observed that for quantized values the BER and FER is higher than the non quantized ones due to quantization errors.

The Verilog outputs also do not give output for the first few cycles before it starts to give the output.

Further improvements can be made by implementing the ldpc layered decoder as it takes far less runtime and memory than the ldpc parallel decoder. Also better quantizers could help us to improve the precision.