A Review on the Performance of Turbo Encoder/Decoder

Ekta Rai[1], Mohd Amir Ansari[2], Mohd Javed Khan[3]
Department of ECE
Integral University Lucknow

Abstract- *In this paper, turbo encoder is designed using the parallel link of two convolution encoder which is isolated by interleaver. Turbo codes assume in pivotal job in various application for instance portable radio, modernized video, remote satellite correspondence, deep space communication, military application et cetera. Here, simulation of turbo encoder and decoder which is parallel link of recursive precise convolution encoder and interleaver are simulated using MATLAB*

Keywords- Interleavers, turbo code, recursive systematic convolution (RSC) encoders.

I. INTRODUCTION

Turbo codes initially present in 1993, can accomplish a bit- error probability of 10^{-5} with an Eb/No of 0.7 dB utilizing a rate 1/2 code over an additive white Gaussian noise (AWGN) channel and BPSK modulation. Turbo Codes are powerful error correcting codes and its performance is very close to the SHANNON limit as it is compared with other class of error correcting codes in general, turbo codes are comprised of two or more component encoders that can be arranged in a variety of ways such as parallel, serial or hybrid concatenation[1][2]. Turbo encoder comprises of two recursive systematic convolutional (RSC) encoders and an interleaver. The encoded data is produced by RSC encoders and is send through the channel, the gotten by an iterative, soft-input/soft- output (SISO) decoding, and the decoded execution can be enhanced as the quantity of interpreting iterations increments. For a reliable and efficient transmission of information, different error correction codes are created. Convolutional codes are created for real-time error correction. Convolutional codes produce one single code word from the whole information bit stream. Turbo Codes are a class of ground-breaking blunder redressing codes, Turbo codes deliver high weight codes by utilizing recursive convolutional encoders, valuable to error correcting codes effectively in the translating procedure.[3]

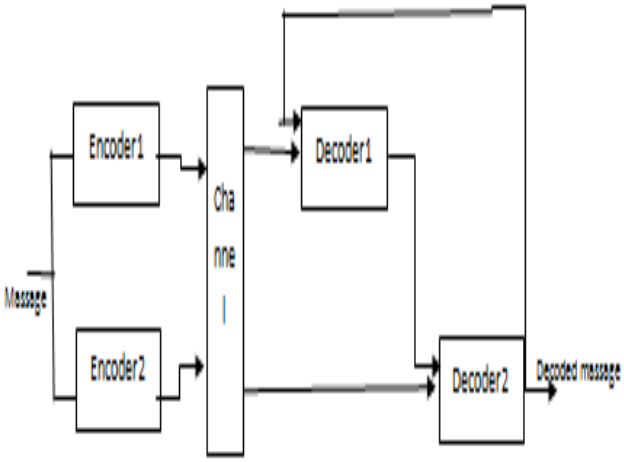

Figure 1: Block diagram of Turbo encoder/decoder

II. THE TURBO ENCODER

RSC (Recursive Systematic Convolution) Turbo Encoder is comprised of 2 Rate ½ RSC encoder as shown in Figure 1.The first encoder takes the input information bits and generates Parity bits P_0. The interleaver π interleaves the information bits X to generate interleaved information Xπ. The second encoder uses Xπ and generates Parity bits P_1.
Turbo encoder consists of two RSC code and interleaver. The generating matrix for RSC can be written as:

$G(D) = [1\ (1+D+D2)/(1+D2)]$

which gives $C_0 = X(D)$
and

$$C_1 = X(D)[1+D+D^2]/[1+D^2]$$

Now assume $F(D) = X(D)/[1+D^2]$

From these relations we obtain

$C_0^{\,j} = X^j$, $C_1^{\,j} = X^j + F^{j-1}$ and $F^{j-1} = F^{j-2} + X^j$

If we consider (F^{j-1} and F^{j-2}) to be current state . The state diagram and Trellis Diagram are obtained is shown in figure 3.

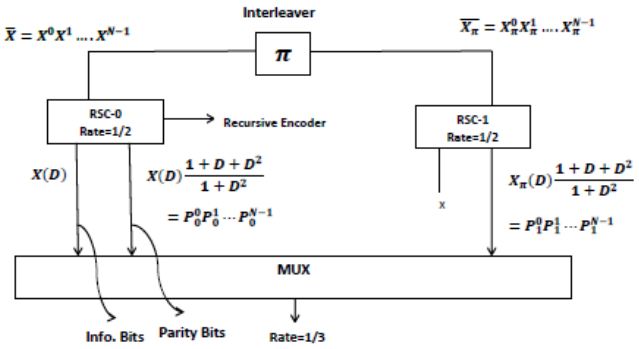

Figure 2: Turbo Encoder

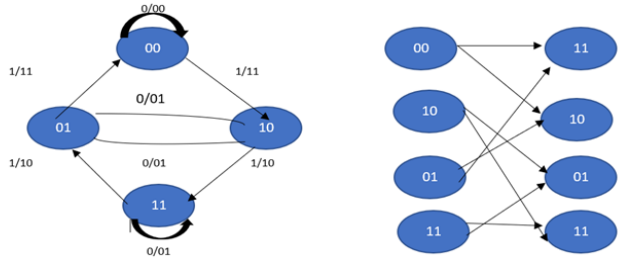

Figure 3: State diagram and Trellis Diagram

Inside the turbo encoder, information is encoded twice through the interleaver. The first RSC encoder has two outputs, one for information bits, v_0 and one for parity bits, v_l.
$V_0= v_0 v_1 v_2 \ldots v_n$ and $P_1= p_0 p_1 .. p_n$
the primary encoder will be prepared by the interleaver and fed into the second RSC encoder.

This one will create at yield just the equality check arrangement. The last encoded grouping is acquired by multiplexing the three yield arrangements from the two encoders. The code rate will be 1/3.Due to the consecutive piece preparing of the interleaver, the turbo encoder can be viewed as a square encoder. The trellis related to the code must be viewed as finished, when it prompts the "0" state. This is fundamental on the grounds that the underlying state for the following square is additionally "0". To lead the trellis into a "0" state, 'v' last bits are presented after the 'n' data bits. The last bits rely upon the encoder state after the data bits are prepared [4].

RSC Encoders creates two unique codes one is efficient yield and the second one is equality bits. In any case, each RSC encoders takes diverse piece stream as info. The first will accept unique information as information and second will take interleaved information as info. RSC Encoders produces two unique codes one is efficient yield and the second one is equality bits. Be that as it may, each RSC encoders take the diverse piece stream as an information. The first will accept unique information as information and second will take interleaved information as info. Every part encoder is isolated from one another by an interleaver. The interleaver has two purposes. Initial, a great interleaver configuration yields codewords having a decent weight appropriation, or, in other words trait nook working at low estimations of E/No [6][8]. Second the interleaver empowers the segment decoders to give uncorrelated a posteriori probability (APP) data for the unraveling of every datum bit.
Transmission of BPSK modulated symbols through AWGN channel: Modulation using BPSK modulation:
$U_0^j=2X^j-1$
$U_1^j=2P_0^j-1$
$U_2^j=2P_1^j-1$
For j=0,1….N-1

Transmission through AWGN channel

$$R_i^j=\sqrt{P}u_i^j+n_i^j$$

For j=0,1,….,N-1 and i=0,1,2. Here P is signal power. Noise variance is assumed to be unity[9][15].

III. THE TURBO DECODER

Most turbo interpreting calculations depend on direct code trellis translating strategies. These are recursive calculations that gauge the information succession. One such calculation is the Viterbi calculation, which limits the blunder probability of the information arrangement. Its yield is a hard estimation of transmitted images that have the most noteworthy probability of showing up in a code grouping. Inside linked frameworks, with various stages for flag handling, delicate estimation creates better outcomes at gathering [10]. A general turbo decoder comprises of two Soft-Input-Soft Output (SISO) processors working iteratively on the got information succession Each decoder registers a LLR for the kth transmitted information bit dk, as

$$L(dk)= \text{Log}[p(dk=1|y)/p(dk=0|y)]$$

where Y is the gotten uproarious succession. The LLR calculations can be performed by either Maximum a posteriori probability (MAP) calculation or Soft-Output Viterbi Algorithm (SOVA). Three metric qualities are required to process a LLR:

1. Branch measurements are ascertained for every conceivable trellis progress

2. Recursive figuring of the forward state measurements

3. Essentially, the retrogressive state measurements are figured by a regressive recursion from trellis time k N = down to k=1

In BCJR (Bahl Cocke Jelinek Raviv) decoder ,there are two decoders, BCJR-0 and BCJR-1. [12][14]

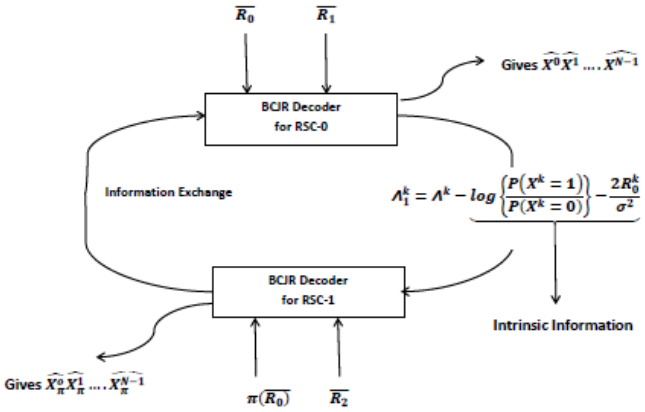

Figure 4: Turbo Encoder

The principal decoder accepts $\overline{R}_0$ and $\overline{R}_1$ as information sources and second decoder takes $\pi\overline{R}_0$ and $\overline{R}_2$ as data sources.
Log Aposteriori Probability Ratio (LAPPR) is defined as
$\lambda^k = \log\{P(X^k = 1|R)/P(X^K = 0|R)\}$
For k= 0,1…N-1.

A noteworthy issue for reasonable utilizations of turbo codes in fast advanced information correspondence is the high dormancy caused by the forward and in reverse state metric calculations. The BCJR decoder needs both the forward and the regressive state measurements and additionally the branch measurements to begin processing LLRs. Another decoder class depends on limiting piece blunder rate probability. some Apriority Probabilities for sources of info 0 and 1 at all stages the main decoder computes LAPPR for all stages and translates they got code bits. At that point it passes the LAPPR esteems barring the characteristic data to the second decoder (As portrayed in Figure 4), which utilizes these LAPPR esteems to compute Priory probabilities at all phases for the two sources of info. What's more, now the second decoder rehashes a similar procedure fused by first decoder. This emphasis stops when decoded bits utilizing both the decoders wind up same or Priory Probability for one of the information progresses toward becoming 1[13][17]. The decoder is made out of two serially connected BCJR decoders connected by an interleaver indistinguishable to the one in the encoder. The decoders utilize a similar rendition of the producing network of the RSC, G, characterized beforehand. The principal BCJR decoder gets the data succession RO and the equality check grouping R1. The decoder delivers a delicate yield that is interleaved and utilized for enhancing the apriori probability gauge for the second decoder. The other two contributions for the second decoder are; the interleaved data succession and the equality check grouping created constantly decoder. The delicate yield of the second BCJR decoder is utilized for enhancing the apriority probability gauge for the main decoder. The quantity of emphasess for this task will enhance the general decoder execution. The negative response circle is the fundamental component ofthe decoder. After a specific number of itrations, the delicate choice will be the equivalent for the two decoders and afterward the deinterleaving will be performed and the hard choice will be taken. Calculation Design After getting transmitted bits at the recipient with Additive White Gaussian Noise (AWGN), we begin with instating apriori probabilities to 0.5 for both information 0 and 1. At that point we compute Y probabilities at all phases for the two data sources. The introduction of α probabilities for first stage is finished expecting encoder to be in state 00 initially. Then we utilize recursive connection of α to figure α of next stage. Also β probabilities are introduced at the last stage i.e. N-1, Where N is square length. In the event that we accept the following encoder to be in state 00 at begin then β can be instated . After computation of α, β, and Y the proportion is figured for all stages. Watching the estimations of λ^K, regardless of whether they are certain or negative the bits are decoded into 1 and 0 individually. After the principal iteration the primary decoder (BCJR-0) passed λ^K esteems to second decoder barring inherent data. The second decoder utilizes these probabilities to compute cloister probabilities at all stages .The second decoder rehashes a similar procedure to disentangle the bits. In reenactment this iterative procedure is ended when Priory Probability for one of the information moves toward becoming 1[14].

IV. SIMULATION

For simulation, I have utilized N=1 million bits of data and shifted SNR from 0 dB to 9 dB. Add up to time taken for 4 iterations and all estimations of SNR going with an augmentation of 0.5 dB (i.e. 19 estimations of SNRs) it took 245.6 sec. Henceforth time taken for each SNR value=12.974 sec. what's more, for every iteration it is 3.24 sec, or, in other words thinking about the expansive number of data bits. When we change the estimation of n then the bit mistake is changed in the event that we increment the estimation of n the bit blunder rate is expanded and when we diminish the estimation of n then the bit mistake rate is diminished.

BER values for all 4 iteration and theoretical bounds on convolution are shown in Table1.

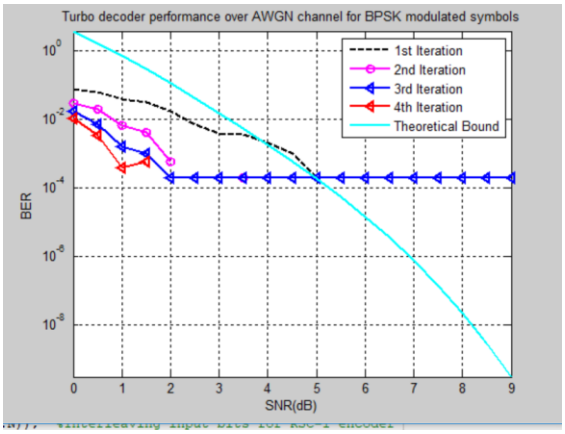

Figure 5: For N = 10^4 Turbo Decoder Performance Between BER And SNR

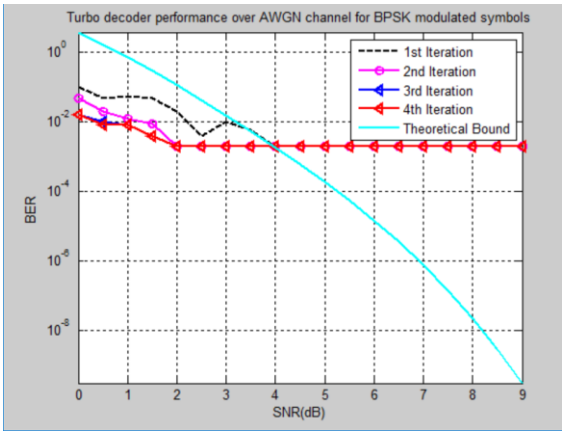

Figure 6: For N = 10^3 BER and SNR Turbo Decoder Performance

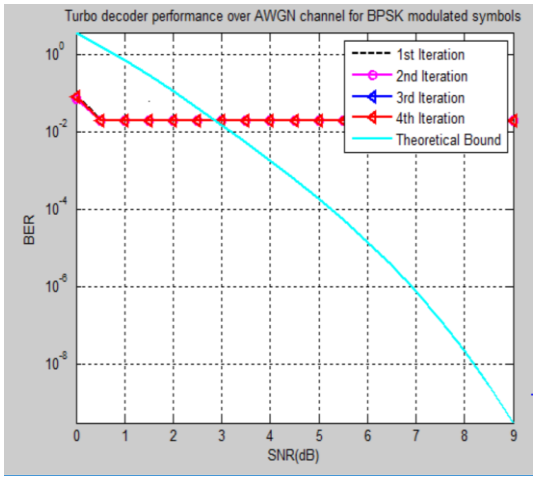

Figure 7: For N= 10^2 TURBO Decoder Performance Between BER and SNR

SNR (dB)	**1st iteration**	**2nd iteration**	**3rd iteration**	**4th iteration**	**Theoretical BER of Convolution code**
0	0.0810	0.0309	0.0182	0.0122	3.4532
0.5	0.05854	0.015	0.007	0.0034	1.6124
1.0	0.0411	0.0059	0.002	$6.6X10^{-4}$	0.7065
1.5	0.02621	0.0022	$5.2X10^{-4}$	$1.1X10^{-4}$	0.2911
2.0	0.0154	$7.54X10^{-4}$	$1.1X10^{-4}$	$2.7X10^{-5}$	0.1134
2.5	0.0074	$1.98X10^{-4}$	$1.6X10^{-5}$	0	0.0422
3.0	0.0046	$4.90X10^{-5}$	0	0	0.0151
3.5	0.0023	0	0	0	0.0053
4.0	$8.18X10^{-4}$	0	0	0	0.0018
4.5	$3.76X10^{-4}$	0	0	0	$5.8542 X10^{-4}$
5.0	$1.56X10^{-4}$	0	0	0	$1.8331 X10^{-4}$
5.5	$5.30X10^{-5}$	0	0	0	$5.3848 X10^{-5}$
6.0	$1.2X10^{-5}$	0	0	0	$1.4566 X10^{-5}$
6.5	0	0	0	0	$3.5593 X10^{-6}$
7.0	0	0	0	0	$7.6999 X10^{-6}$
7.5	0	0	0	0	$1.4434 X10^{-7}$
8.0	0	0	0	0	$2.2907 X10^{-8}$
8.5	0	0	0	0	$2.9997 X10^{-9}$

9.0	0	0	0	0	3.1504 $X10^{-10}$

Table1: Comparison of BER

V. CONCLUSION

From the recreation results for BER for Turbo code, we can see that execution of BCJR iterative decoder is greatly improved than the translating utilizing Viterbi calculation for the convolution code. BCJR calculation can disentangle the transmitted bits with 0 probability of mistake even at high SNR. In spite of the fact that the time unpredictability might be a factor on account of huge number of bits, as a lot of trellis counts are included. To enhance the execution of BCJR disentangling regarding time many decreased state BCJR calculations, for example, BCJR-M and BCJR-T are proposed, which considers normal number of live states per trellis and diminishes computation. Subsequently Turbo code is better as far as execution and time intricacy than different codes. Turbo codes are iterative codes and the execution enhances with the expansion in the quantity of emphases. At the point when the flag control is expanded, the SNR increments. Accordingly the from the earlier data of the information accessible progresses. As the contribution to the decoder currently contains lesser blunders, the decoder can even now yield a message that bodes well in this manner, bring down BER can be accomplished.

The execution of the turbo code significantly relies upon the irregularity presented by the interleaver. An expanded interleaver estimate K results in higher probability of higher weight code words, in this manner bringing about better decoder execution. BER can be accomplished by keeping the SNR steady yet at the expense of expanded dormancy. Additionally, the code can accomplish much lower BER at a predefined SNR over an AWGN channel when contrasted with Rayleigh blurring channel.

Along these lines, we can presume that in planning turbo codes there is an exchange off between vitality effectiveness, data transfer capacity productivity, dormancy, unpredictability and mistake execution.

Turbo codes are being utilized in 3G and 4G portable communication, WiMAX and satellite correspondence frameworks. They indicate remarkable execution at low SNRs moving toward as far as possible. Anyway at higher SNRs, the BER bend starts to level and ruins the capacity to accomplish to a great degree little piece blunder rates. The reenactment results demonstrated that the execution of the turbo code relies upon various parameters including the casing size K, number of decoder emphases.

REFERENCES

[1] Donghoon Kang and Wangrok Oh, Member, IEEE, " Faster Than Nyquist Transmission With Multiple Turbo-Like Codes ", IEEE Communications Letters, Vol. 20, No. 9, pp.1745-1747, Sep-2016.

[2] Ertan Ozturk, Hakan Kaya," Performance analysis of distributed turbo coded scheme with two ordered best relays ", IET Communications , Vol. 9, Iss.5, pp. 638–648, 2015.

[3] Tepoju Vivek , Vardhan Bandi, Neeraja Boya, Pradeep Kumar, Chandra Sekhar Paidimarry., " Implementation Of Turbo Codes Using Verilog-Hdl and Estimation Of its Error Correction Capability ", IEEE Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics, pp. 75-79, 2015.

[4] C. Berrou, A. Gliavieux, and P. Thitimajshima., " Near Shannon limit error-correcting coding and decoding: Turbo-codes ", IEEE International Communication Confrence (ICC), pp.1064-1070, May 1993.

[5] C. Roth, S. Belfanti, C. Benkeser and Q. Huang, "Efficient Parallel Turbo-Decoding for High-Throughput Wireless Systems", IEEE Transactions On Circuits and Systems—I:,vol. 61, no. 6, Page:18241835, June 2014

[6] M. Martina, S. Papaharalabos, P.T. Mathiopoulos, and G. Masera, "Simplified Log-MAP Algorithm for Very Low-Complexity Turbo Decoder Hardware Architectures", IEEE Ttransactions on Instrumentation and Measurement, vol. 63, no. 3, pp: 531-537, Mar 2014.

[7] L. Li, R.G. Maunder, B.M. Al-Hashimi and L. Hanzo, "A LowComplexity Turbo Decoder Architecture for Energy-Efficient Wireless Sensor Networks", IEEE Transactions On VLSI, vol. 21, pp: 14-22,2013

[8] L. Li, R.G. Maunder, B.M. Al-Hashimi, M. Zwolinski, and LajosHanzo, "Energy-Conscious Turbo Decoder Design: A Joint Signal Processing and Transmit Energy Reduction Approach", IEEE Transactions on Vehicular Technology, vol. 62, no. 8,pp:3627-3638 Oct 2013.

[9] N. Abughalieh, K. Steenhaut, B. Lemmens and A. Nowe, "Parallel Concatenation vs. Serial Concatenation Turbo Codes for Wireless Sensor Networks", IEEE Symposium on Communications and Vehicular Technology in the Benelux (SCVT), pp-1-6, Nov 2011

[10] P. Reddy, F. Clermidy, R.A. Khayat and A. Baghdadi, "Power Consumption Analysis and Energy Efficient Optimization for Turbo Decoder Implementation", IEEE Int Symposium On System On Chip, pp:12-17, 2010.

[11] M. Martina, and G. Masera, "Turbo NoC a Framework for the Design of Network-on-Chip-Based turbo Decoder Architectures", IEEE Transactions on Circuits and Systems,vol. 57, no. 10,pp:2776-2789 October 2010.

[12] S. Shah & V. Sinha, "Iterative decoding vs. Viterbi decoding:a comparison", in Proc of National Conference on Communications, IIT Bombay, pp. 494-497, February 2008.

[13] I. Atluri, A. K. Kumaraswamy and V.A Chouliaras, "Energy efficient architectures for the Log-MAP decoder through intelligent memory usage", IEEE Computer Society Annual Symposium on VLSI,pp-263265, 2005 .

[14] Shobha Rekh, S. Subha Rani, A. Shanmugam, "Optimal choice of interleaver for turbo codes", Academic Open Internet Journal, volume 15, 2005.

[15] Muhammad Imran Anwar and Seppo Virtanen and Jouni Isoaho: "a software defined approach for common baseband processing", Journal of Systems Architecture, Jan. 2008.

[16] S. Choudhury, "Modeling and Simulation of a Turbo Encoder and Decoder for Wireless Communication Systems," UT Austin, 2002.

[17] D. Divsalar and F. Pollara, "Turbo codes for PCS applications", Proceedings of ICC 1995, Seattle, Washington, pp. 54-59, June 1995.