Design and Analysis of Synchronous TurboCodec Using Randomizer on FPGA

Srikanth R Vijaya Prakash A. M

MTech in VLSI and Embedded System, BIT Bangalore Professor, Dept of ECE, BIT Bangalore

***Abstract*: Among forward error correcting codes, Turbocoding is a potent error correction method. It performs wellbecause it gets close to the Shannon limit. The Turbo encoderand decoder are designed using the Turbo codec, which ispresented in this work. The Viterbi algorithm, whichincorporates hard-input and hard-output values, is the basisfor the development of the decoder. In order to estimate errorcorrecting capability, errors are intentionally introduced intothe encoded data. In this situation, the newly created Turboencoder and decoder are more synchronous to each other byeliminating long strings of zero's in the encoded data, this willbe achieved by scrambling the input data bits by Randomizer.Which is also introduce more randomness in the encoded datawithout altering the code rate of the encoder, this will reducethe latency of the encoder and decoder. Verilog-HDL is used inthe Turbo codec's encoder and decoder implementation.Almost in every communication applications, the turboencoder and decoders are using to encrypt and decrypt themessage bits with error correction capability, for exampleSDR's (Software Defined Radio's). To increase the morerandomness in the code and also to increase the synchronization at the decoder end, here we proposing theconcept called scrambling the bits (also called Randomization).**

***Keywords*: FPGA, HDL, Recursive Convolutional Encoder (RSC), Viterbi algorithm, Randomization, SDR, Scrambling, Xilinx.**

INTRODUCTION

The wireless technology revolution has altered how we communicate. Individuals can communicate with other people and places on the planet from anywhere by using cellular and satellite technology. We must deliver information fast and precisely as we become increasingly reliant on technology. Voice and data should be sent usingtransfer mistakes since wireless systems' communication channels are substantially more hostile than those of "wired" systems. Information will be sent using corrective coding to lessen the possibility that channel effects will destroy it. The next level of coding can be reached using anew technique called turbo coding. Performance that is closer to the bounds of theory than conventional coding. Compared to wired systems, wireless systems have highererror rates in voice and data transmission via channels. Tolessen the likelihood of errors caused by information transmission channel corruption, forward error correcting codes should be utilized. Because it offers a high level of performance, turbo coding, an effective coding technique, is favoured above conventional coding techniques. The 3G standard has been turbo charged. Turbo codes aredistinguished by their use of interleavers, recursive systematic encoders, and iterative decoding methods. Turbo code increases the effectiveness of data transfer in digital communication systems. The theoretical examination of polynomial selection, weight distribution imposed by interleaver design, decoder error floors, and iterative decode thresholds served as the foundation for the invention of turbo codes. The turbo cord family has been standardized, put into practise, and is now utilized ona number of spacecraft. Near the Shannon limit, turbo codes allow for dependable communication over power- restricted communication routes. However, getting thisoutcome needs a lot of iterations, which increases latency.In order to detect faults and/or repair them in the presence of channel noise, channel coding encrypts the data sent through the communication channel. Therefore, an ongoing field of research is the effective implementation of turbo codes to fulfil real-time restrictions. Without altering the code rate of the turbo codec, the more randomness in the code can be achieved by scrambling themessage bits. This will improves the error capability of the decoder and also increases the synchronization between encoder and decoder by eliminating long strings of zero's in the encoded data bits. Nowadays security is more crucial parameter while transmitting the data. To achieve this parameter, there is a need for encoding the data, this can be done by the Turbo encoder. To increase the randomness in the code, need to vary the code rate. If we vary the code rate the latency will also increase. So the encoding and decoding operational speed will decrease and also, there may be a low synchronization between the encoder and decoder dueto long strings of zero's in the encoded data. It may affect the error capability of the decoder.

OBJECTIVES OF THE PROPOSEDMETHODOLOGY

I. The Randomizer will eliminates the long strings of zeroes in the encoded information bits to provide moretransitions in the data. This helps receiver for synchronization to recover the original bit pattern.

II. It will increase the randomness in the data without increasing code rate unlike block coding technique.

III. It improves error detection capability.

PROPOSED METHODOLOGY

The below diagram, as shown in figure-3 represents the flow of the proposed methodology. The flow of the design starts with the randomizer, which scrambles the incoming information bits and the its fed to the turbo encoder followed by data assembler. At the decoder part, there will be a data de-assembler, which segregates the systematic bits,parity -1 and parity-2 bits, which are going as input to the

turbo decoder based on viterbi algorithm and finally decoded bits are derandomized and recover the original information bits. Each block of the proposed design as discussed in the below sections.

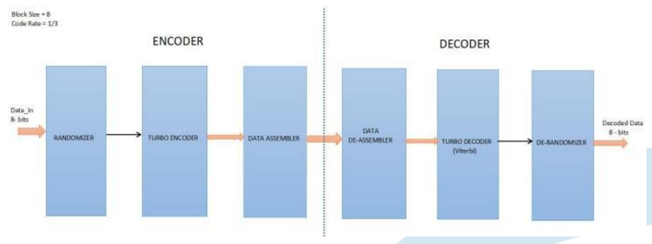


Figure 3:Methodology Flow Diagram

1. Randomizer

The first block in the flow diagram will scramble the incoming data bits to eliminate the long strings of zero's in the message bits. This process is called randomization. Which helps to improve the error capability of decoder. The design of randomizer is based on D-Flip-Flop and some XOR gate, which is basically modulo-2 addition of incoming bits and the previous bits in the registers.

Turbo Encoder

The fundamental turbo code encoder is constructed by concatenating two identical recursive systematic convolutional (RSC) encoders in parallel. The two parameters most frequently used to characterize convolutional codes are the code rate (r) and constraint length (K). The turbo encoder's output is measured by the ratio of the number of channel symbols it produces to the bits it receives and represent the coding rate, which is expressed as a ratio in a specific encoder cycle. Here, the block size is 8 and code rate is 1/3, so for one block there will be 24 bit encoded data, which are systematic bits, parity-1 and parity-2 each of 8-bits. The interleaver is on matrix based, which basically re-arrange the bit position in the block. The interleaver ouput will be fed to the RSC2.

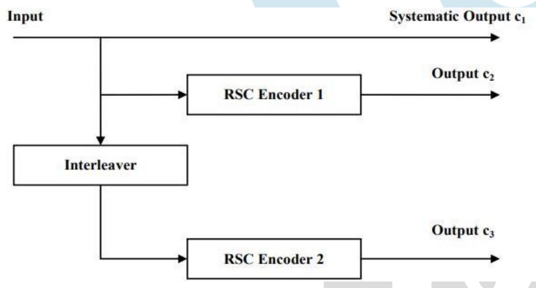


Figure 5.1: Turbo Encode (Code Rate r = 1/3)

5.1 RSC Encoder

The two D-flip-flops and two XOR gates that make up theRSC Encoder are used. The RSC encoder block receives an input bit stream with a length of 8 bits. Following the RSC Encoder's state structure, which is depicted in Fig. 5.1, it encodes the bit stream in a specific order.

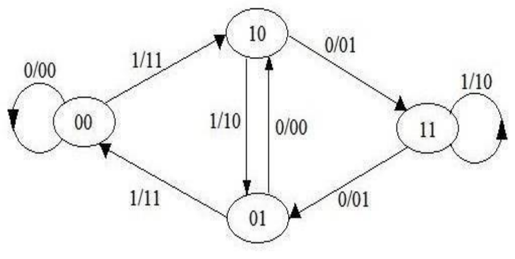
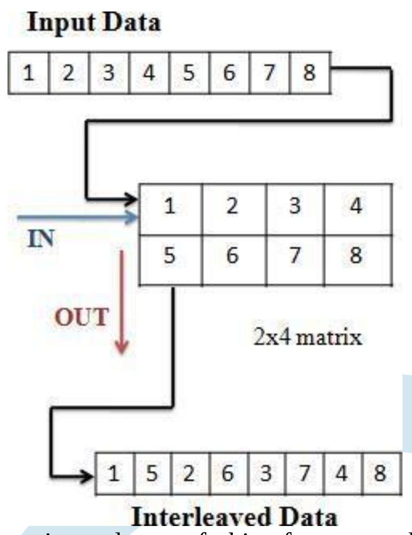


Figure 5.2: State Diagram of RSC Encoder

Encoding initially begins at state "00." The encoder's output is derived from the input and advances to the subsequent state along that branch. In a similar manner, it switches between states to create the output bits for thematching input bits.

5.2 Interleaver

Block Interleaver writes data in a two-dimensional array row-wise from left to right and top to bottom. Following data entry



into the array, the data is read out in a columnar fashion from top to bottom and left to right. The blockinterleaver uses an array that is 24 in size.

Figure 5.3: Matrix based Interleaver

The RSC2 Encoder receives the output of the interleaver as input. In Fig. 4, the block interleaver is displayed. The placements of the information bits are indicated by the numerals in Fig. 5.3.

5.3 Data Assembler

The data assembler will assemble the 24 bit encoded data for single block with code rate 1/3 and also it generate one valid pulse for every block of encoded data. The implementation of data assembler is based on serial in and parallel out methodology using some registers. The encoding bits systematic, parity-1 and parity-2 bits are shifted by left into the pre-declared register of 24 - bits. Once the encoding of block size 8 is done. It will generate the 24 bits of encoded data with valid pulse indicating that, the encoded data is valid.

Turbo Decoder

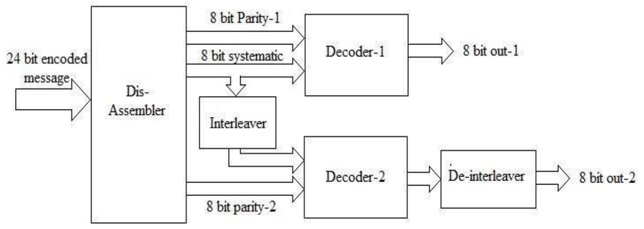
The turbo decoder is based on the viterbi algorithm, The Turbo Decoder is made up of Data Dis-assembler, Interleaver, two Decoders, and De-interleaver, as seen in Fig.
6.1. The Data De-assembler block is originally provided the encoded data that was received.

6.1 Data De-assembler

Then data de-assembler will segregates the encoded dataof 24 bits based on the valid, which are systematic bits, parity-1 and parity-2 and fed to the decoder. The data de- assembler is exactly opposite to the data assembler functionality. The incoming encoded bits of 24 with valid pulse are received by data de-assembler and based on valid pulse the 24 bits of encoded data will be segregates into the three signals, which are systematic bits, parity-1 and paruty- 2 bits. These signals are next fed into the turbo decoder.

6.2 Viterbi Algorithm based Turbo Decoder

In order to create turbo decoder, two algorithms— maximum a posteriori (MAP) and soft-output Viterbi algorithm—are frequently utilised (SOVA). Bit Error Rate (BER) performance is often greater for turbo decoders usingthe MAP algorithm, but their complexity and latency are very high. The delay issue can be solved by MAP decoders using the sliding window (SW) approach, although theircomplexity is still very high.
Turbo decoders based on SOVA can, however, be developed with greater performance and less complexity. However, compared to MAP decoders, its BER performanceis not noticeably worse. Therefore, SOVA approach is favoured over MAP decoder. In SOVA, soft decision decoding is used, where the received signal is effectively determined as either "0" or "1" and is represented as either strong or weak dependent on signal strength. In order to simplify things, Hard Decision Decoding is employed in this paper. The signals in Hard decision are denoted by the single digits "0" or "1". Compared to hard decoding, soft decoding



requires more modules and is more sophisticated.

Figure 6.1: Turbo Decoder

The encoded bit stream is split into 3 bit streams by the data disassembler: input, parity-1, and parity-2. The Decoder-1 block receives the eight - bit input and the parity-1 as inputs. Parity-2, or interleaved parity, is one of the inputs for the Decoder-2 block. Therefore, the interleaved input again for Decoder-2 block should output an interleaved version of the actual input data. De-interleaver can be used to extract the actual information by this interleaved data.Two similar Decoders make up the Turbo Decoder, whichemploys the Viterbi algorithm for decoding. The example that follows explains Viterbi decoding. Consider that the received message is in Table 1 because the original encoded message was distorted. The bits that were received in red are corrupted bits. The decoder is responsible for fixing these mistakes. According to Fig. 6.2, the critical path in a Trellis structure is dependent on the data bits that were received.

Table 1 : Encoded and Received Data

Encoded :	11	01	00	10	01	10	11
Received :	11	11	00	10	01	11	11

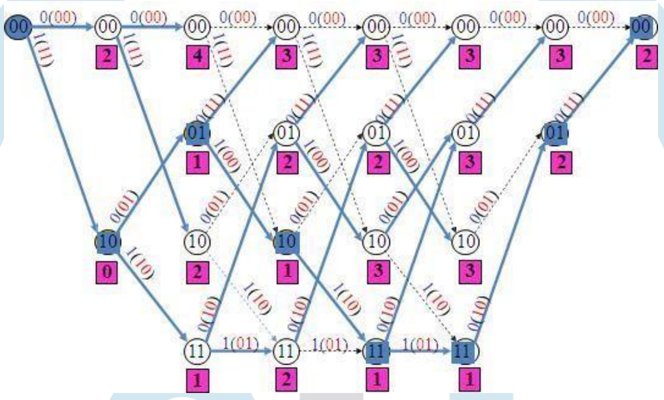


Figure 6.2 : Trellis Network of Viterbi Algorithm

The Code trellis arrangement, which is depicted in Fig. 6, is crucial in the Viterbi algorithm. Trellis code originates in the state "00" and terminates in the same "00." The viterbi decoder only accepts single bit of input and one bit of parity at a time. With branch, these two bits are compared the values from Fig. 6.2 are placed in brackets, and the following state calculates the Hamming distance. Each stage's hamming distance is displayed in pink colour boxes. The Hamming distances are calculated at each branch and added to the distance at the state of origin at the following branch. The path with the greatest Hamming distance isremoved at each level, and the path with the smallest Hamming distance is ultimately chosen. The output of the decoder is represented by the bits on this path. This decoder output represents the information the most accurately. Interleaver and De-interleaver work in a complimentary way. It employs the input data and writes it in a two- dimensional array of order 2X4 column-wise through top to bottom and left to right. Once entering the data into the array, it is read out row-wise from top to bottom and left to right, as depicted in Fig. 6.2.

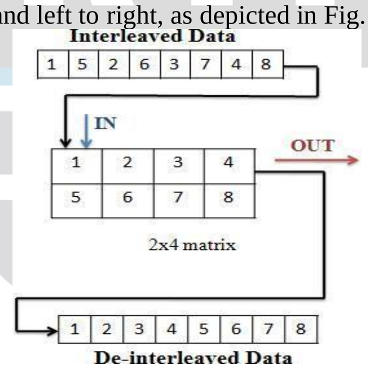


Figure 6.3: De-interleaver

6.3 De-randomizer

The De-randomizer will get input as decoded data, the Design of the de-randomizer will be complimentary to the Randomizer. The decoded data from the decoder will not be the original data. Initially the data bits are scrambled by the Randomizer. So, now the srambled bits are De-srambled by the De-randomizer. The output of the de- randomizer will reconstruct the original data bits.

RESULTS AND DISCUSSION

The turbo encoder and decoder are designed and simulated in Xilinx's Vivado 2018.2.

7.1 Turbo Encoder

The turbo encoder will receive the 8 bits of parallel data and which converts to serial data internally. This serial data is fed to the RSC encoder 1 and interleaved output is fed to the RSC encoder 2. Totally 24 bits of encoded data will be obtained from the encoder for the block size 8.

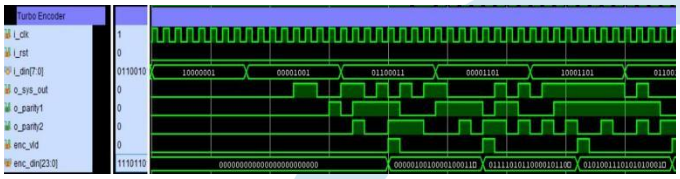


Figure 7.1: Simulation Results of Turbo Encoder

The simulations of the designed Turbo encoder as followsas seen in the following snapshots in figure 8. Here, the 8-bitdata "10000001" given as input to the turbo encoder and the respective encoded data will be generated as "000001001000010001111010". The encoded data will be appended as systematic bits, parity1 and parity2 from the LSB.

7.2 Comparison of Turbo Encoder

This comparison is between Turbo encoder output with Randomizer and without Randomizer.

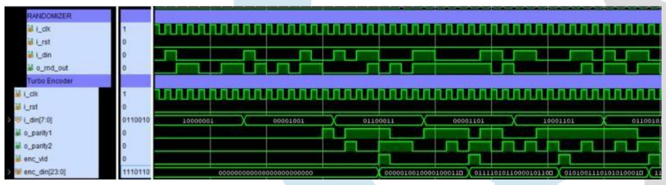


Figure 7.2: Simulation Results ofTurbo Encoder withRandomizer

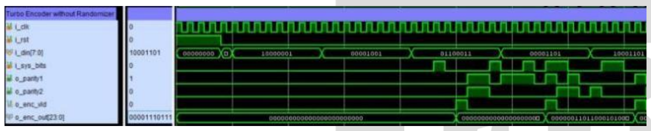
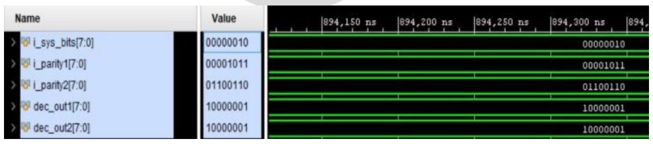


Figure 7.3: Simulation Results of Turbo Encoderwithout Randomizer

The above wave-forms shows the comparison between with and without randomizer. Here, the input is same for both the encoders as 8-bit data of "10000001". The encoded data with the randomizer is generated as 24-bits of "000001001000010001111010". But for another encoder without randomizer with the same input of "10000001" as generated the encoded data, which is 24-bits of "000000000000000000000001". Now, its clearly show that, the long strings of zero's will be eliminated by scrambling the input data bits by using Randomizer. Which increasesthe randomness in the code and also improves the synchronization between encoder and the decoder to decode the original bits of information.

Turbo Decoder :



```
Name              Value      |894,150 ns |894,200 ns |894,250 ns |894,300 ns |894,
i_sys_bits[7:0]   00000010                                          00000010
i_parity1[7:0]    00001011                                          00001011
i_parity2[7:0]    01100110                                          01100110
dec_out1[7:0]     10000001                                          10000001
dec_out2[7:0]     10000001                                          10000001
```

Figure 7.4: Simulation Results of Turbo Decoder

The figure-11 waveform shows the decode data. Here, intentionally errors are introduced in the encoded data to verify the functionality of the decoder. For the encoded data of 24-bits "000001001000010001111010", the decoded datais "10000001", which is same as encoder input.

7.3 RTL Schematic

The figure- 7.7 shows the RTL schematic of the Turbo Codec, which is implemented on Xilinx’s Vivado 2018.2. The targeted FPGA board is ZedBoard Zynq Evaluation and Development Kit (xc7z020clg484-1). Which has 6 input LUT structure of CLB’s (Configurable Logic Blocks).

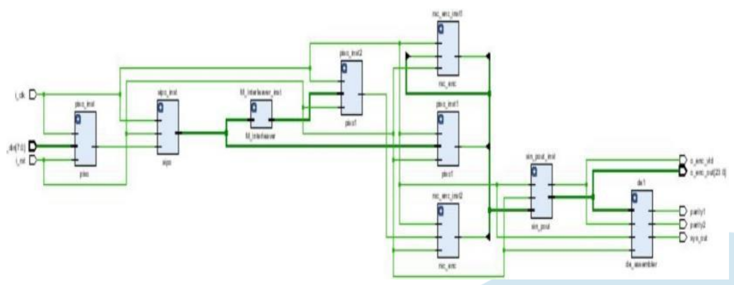


Figure 7.5: RTL Schematic of Turbo Encoder withoutRandomizer

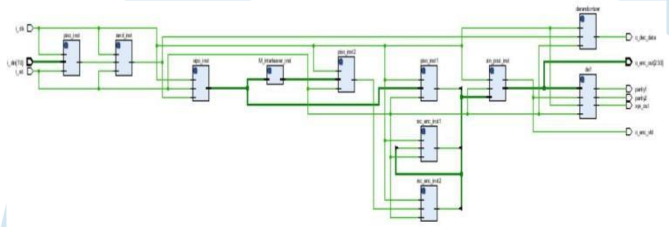


Figure 7.6: RTL Schematic of Turbo Encoder withRandomizer

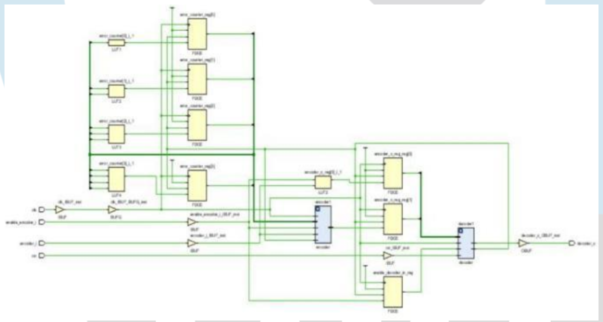
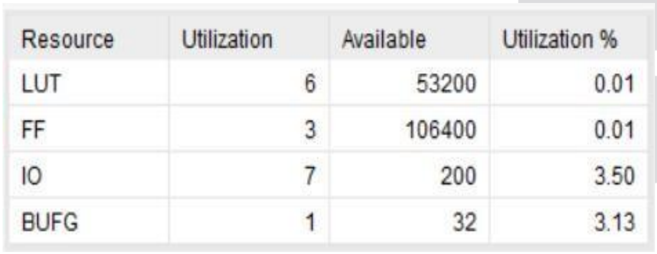


Figure 7.7: RTL Schematic of Turbo Codec

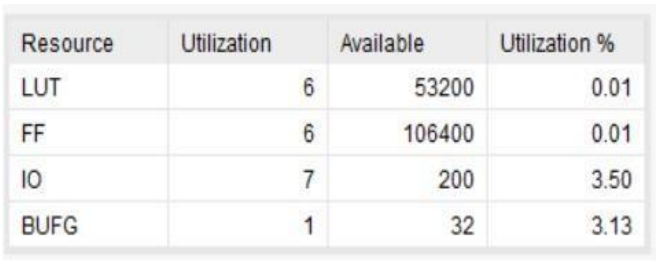
7.4 Resource Utilization

Table 2: Utilization Report of Turbo Encoder withoutRandomizer



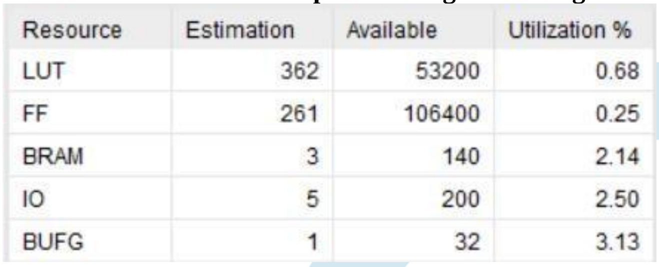
Resource	Utilization	Available	Utilization %
LUT	6	53200	0.01
FF	3	106400	0.01
IO	7	200	3.50
BUFG	1	32	3.13

Table 3: Utilization Report of Turbo Encoder withRandomizer



Resource	Utilization	Available	Utilization %
LUT	6	53200	0.01
FF	6	106400	0.01
IO	7	200	3.50
BUFG	1	32	3.13

Table 4: Utilization Report of Integrated Design



Resource	Estimation	Available	Utilization %
LUT	362	53200	0.68
FF	261	106400	0.25
BRAM	3	140	2.14
IO	5	200	2.50
BUFG	1	32	3.13

The table -4 shows the resource utilization for the Turbo Codec, Look-up tables(LUT) are 362, Flip-Flop's(FF) are 261, Block RAM's are , which are 18KB's each, Input andOutput (IO) are 5 and finally one BUFG to drive the clock.

CONCLUSION

This project mainly focuses on the simulation of the design of the Turbo Codec for a better more realistic comparison with and without Randomizer. The Turbo Encoder and Decoder are designed along with the randomizer to scramble the information bits to increase the synchronization at the receiver and simulated in Verilog- HDL using Vivado for 8 bit input. Randomized bits are given to the Turbo Encoder and encoded bits are obtained. These encoded bits are transmitted to the Turbo Decoder through channel. At the Turbo Decoder, the received data may contain errors, which are decoded using Viterbi algorithm to obtain the original information.The Turbo Codec proposed can also be extended to other code rates like 1/4, 1/5, 1/6 so on,based on the application and the latency can also be reduced by using multiple instances of the same block, for example, the current design as latency 0f 21usec for block size of 8. Similarly, with the same latency, block size of 32 will also be encoded anddecoded by four instances.

REFERENCES

[1] C. Berrou, A. Gliavieux, and P. Thitimajshima., "Near Shannon limit error-correcting coding and decoding: Turbo-codes." *IEEE* International Communication Confrence *(ICC),* Geneva, pp.1064- 1070 2019.

[2] A. Burr, "Turbo-codes: the ultimate error control codes?", ELECTRONICS & COMMUNICATION ENGINEERING JOURNAL AUGUST 2018 ,pp. 155-165.
.

[3] Han, J.H.; Erdogan, A.T.; Arslan, T., "Implementation of an Efficient Two-Step SOVA Turbo Decoder for Wireless Communication Systems" Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE, Year: 2020, Volume: 4 pp.2429-2433.

[4] J. Hagenauer and P. Höher, "A Viterbi Algorithm with Soft Outputs and It's Application," in Proc. IEEE Global Commun. Conf., pp. 47.1.1- 47.1.7, Nov. 2019.

[5] Hamid R. Sadjadpour, *Senior Member, IEEE*, Neil J. A. Sloane, *Fellow,IEEE*, Masoud Salehi, and Gabriele Nebe, "Interleaver Design for Turbo Codes", IEEE JOURNAL on selected areas in Communications, Vol. 19, NO. 5, MAY 2016 pp. 831-83.

[6] M.C. Valenti, "Turbo codes and iterative processing," in *Proc. IEEE New Zealand Wireless Commun. Symp.*, (Auckland, New Zealand), Nov. 2015.

[7] V. Kavinilavu1, S. Salivahanan, V. S. Kanchana Bhaaskaran, Samiappa Sakthikumaran, B. Brindha and C. Vinoth, " Implementation of Convolutional Encoder and Viterbi Decoder using Verilog Hdl" in IEEE tran. On inform theory, 2011.

[8] C. Berrou and A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo- codes," IEEE Transactions on Commmunications, Vol. 44, No. 10, pp. 1261- 1271.

[9] D. Divsalar and F. Pollara, "On the design of turbo codes," The Telecommunications and Data Acquisition Progress (TDA)

progress Report42-123, Jet Propulsion Lab (JPL), Pasadena, CA, pp.99-121.

[10] D. Divsalar and R. J. McEliece, “Effective free distance of turbocodes,” Electronic Letters,Vol. 32

[11] M. S. C. Ho, S. S. Pietrobon, and T. Giles, “Optimising the constitute codes for turbodecoders,” International Symposium onInformation Theory ISIT 2020, Cambridge, MA, USA.
.

[12] S. Benedetto, R. Garello, and G. Montorsi, “A search for good convolutional codes to be used in the construction of turbo codes,” IEEE Transactions on Communications Vol. 46, no. 9, Sep. 1998, pp. 1101-1105.
.