**IMPLEMENTATION OF LONG TERM EVALUATION BASED TURBO COMMUNICATION SYSTEM USING MAP APPROACH**

**1. 1. Overview**

Data transfer between systems is critical in today's world, as technology advances and the number of users simultaneously grows. There are serious problems in the digital communication networks because of this widespread use, which results in data corruption. Providing a solution to the faults that arise in the communication process is critical to reducing data corruption in telecommunications. In order to successfully decode and fix a process, the Turbo algorithm is a useful factor to determine. The Turbo method is the most widely recognised algorithm for decoding convolution codes. Software and hardware implementations of this algorithm are possible. The digital systems present data in an effective manner in order to facilitate well-organized communications. Digital communication systems must contend with the problem of data corruption. Error correcting codes are the greatest way to prevent data corruptions. Because of its ability to decode efficiently, it was adopted by virtually all communication systems; even the Turbo algorithm necessitates extremely standard gear. The working obstacles can be removed while the decoding operation is in advance, allowing for the employment of an enhanced approach, the Adaptive Turbo Algorithm. Because this algorithm is so good at handling high-speed operations, it can decode codes in a small-time frame. AVA employs a maximum-likelihood decoding approach that makes use of convolution codes to discover viable code sequences.

In this case, Verilog HDL, which stands for Verilog Hardware Description Language, is employed as a hardware description language for this project. Applied to semiconductor and electronic design, as well as analogue and mixed-signal circuits, this language is used to create electronic systems. ModelSim and XILINX-ISE are the two primary tools used in this study to achieve its goals. These studies provide a thorough explanation of the Turbo Algorithm's execution process, as well as several languages and methods for evaluating its performance.

**1.2. Aim and Objectives:**

**Aim:** Execution of Turbo algorithm applying VHDL coding.

**Objectives:**

• To clearly understand the Hidden Markov model and Turbo encoder.

• To evaluate the basic functionalities and steps involved in Turbo algorithm

• To research on the implementation of Turbo algorithm through VHDL code

• To critically analyze the results obtained through VHDL code.

**2. LITERATURE REVIEW**

Turbo codes are among the most recent and cutting-edge methods for FEC (FCC). [1] [2] The capacity of the channel was increased by using these codes [1] Turbo codes are used in 3G and 4G wireless communications. When data corruption noise is present, they are also utilised in deep space satellite communications and in other situations requiring accurate information to be transmitted via bandwidth or latency constrained communication systems. [3Codes close to channel capacity were being designed as early as 1948. This meant that the decoding difficulty would be extremely high if there was a very random-like code that had no structure. As a result, the difficulty is to build a code with enough randomness and structure such that it may be used for decoding the code. Because of an intrinsic interleaver structure in the parallel composition structure, the turbo codes are a type of codes that are completely random. It's possible to decode these error-correcting codes with ease because of the code's structure.

Parallel concatenation is used to encode a turbo code. Figure 1 shows a traditional block diagram that describes the Turbo encoder better. With the input message marked by u, the turbo codes use a rate-half-convolutional coding. To rate half-convolutional coding, the output of the interleaver block is fed. The systematic version is maintained in the upper block, when the parity is retained in the lower block. Parity bits are used to generate terminations at the responses [4]. As a result, the code's speed is reduced by a third. The rate will be reduced by around a third if the termination is taken into consideration. For relevant results, k should be at least 50 or 100 [5]. [5] Puncturing would be utilised to obtain higher rate codes like rate half. Based on the intended rate, there are a variety of puncturing patterns. Keeping all the odd-numbered bits in one parity and all the even-numbered bits in the other is the normal puncturing pattern for rate half. Both of these blocks must have recursive systematic convolutional encoders, which is critical. Systematic convolutional encoders were used in the early stages of the suggested turbo codes. A number of non-systematic turbo codes have since been devised, although the ones proposed by Berrou et al. in 1993 were based on systematic convolutional transmitters. The usage of recursive encoders is another critical consideration. Feedback encoders are used in the recursive encoders. There is a feedback loop between the input and output. There is a feedback loop between the output and the input, and this feedback loop is mirrored. The greater the minimum distance, the better the code's capacity to repair errors. It is the parallel concatenation, rather than the serial concatenation, of several encoders that makes up the primary component of a turbo encoder. First encoder output bits are provided as input into second encoder in a serial concatenation arrangement. Instead, the interleaver just flips the bit. As a result, bits that were formerly in bit location 1, for example, may now be in bit location 56, resulting in a random assortment of bits. Design and function of the interleaver are now critically important for turbo codes.

3. **ABOUT THE PROPOSED SOLUTION**

**3.1. Overview**

The concept "research" refers to any human activity involving logic and reasoning to discover new information about the world around us. The researcher can do their assignment successfully in various ways, but they must first determine their research goals before deciding on an approach [17]. Secondary materials are preferable to primary sources for this study. Because the researcher may run across difficulties when trying to collect accurate information.

**3.2. Language used for Turbo algorithm**

Many functional programming languages exist because the majority of hardware programmes are written in a hardware description language like VHDL (Very High Speed Integrated Circuits) which cannot be coded in imperative languages like C or MATLAB. Isolated sign language Turbo algorithm and continuous sign language Turbo algorithm are two of the most commonly used methods for searching the frame at the same time. The Turbo algorithm, a dynamic programming algorithm, is used to determine the series of hidden states known as the Turbo path.

The Turbo algorithm relies on the assumption of a state machine. When a system is being represented, it is in one of a limited number of states at any one time. An alternate path that is at least one of many possible paths to a given state when a number of paths can be taken. The method keeps track of the most likely state by reviewing all potential states, which is the underlying assumption of the process. Thus, it is only essential to maintain track of a single path and not the entire track of all the states. Here's the first assumption. It is assumed that a new path from the previous state is marked by additive metric (the second assumption). Another assumption is that events accumulate across time in a dynamic environment. When an event occurs, the algorithm moves forward in a new state, integrating the additive metric with the previous path and evaluating a new set of conditions. The additive metric is associated with the transition property from one direction to the following [18]. Let's look at an example of this in action. It is only possible to transmit data in halves, with half of the symbols coming from even-numbered paths and the other half from odd-numbered approaches.

State transition graphs are almost always incomplete. For the purpose of discovering the Turbo path, a dynamic programming algorithm, the Turbo algorithm is utilised. The Turbo algorithm relies on a state machine assumption. There are only a limited number of states in which a system can exist at any given time. Among the several possible routes leading to a particular condition, the survivor path is one of the most likely.

The 16-state rate 1/2 convolution coder used in the Turbo coder has the following system equations:

G0 (n) = x (n) + x (n-1) + x (n-3) + x (n-4)

G1 (n) = x (n) + x (n-2) + x (n-3) + x (n-4)

Where x (n) is the un-coded input and G0 (n), G1 (n) are the encoded outputs

A 16-state trellis diagram will be used to create the Turbo decoder. Using the C54x DSP's specific instruction set is now possible.

**4. PURPOSE SOLUTION OF TURBO CODING**

**Proposed solution for the problem: Turbo Algorithm**

Wide range applications of the Turbo algorithm are towards the DNA analysis, speech appreciation for cell phones communication and facilitates. The outcome of backtracks from all the branches may obtain the algorithm task. The Turbo algorithm can perform step-by-step function as illustrated:

1) Initialization: Arrange all metric in the perfect format.

2) Computation step j+1: Suppose the previous step and use to identify the basic survivor paths for storage in allthe states.

3) Final step Continue to compute the entire pending algorithm reaches with all-zero state like hood paths. Turbo algorithm is most likelihood detected sequence with the MLSD with in all the inter-symbol interference (ISI) as well as memory less noise considering all the input state channel as well as observable sequence .

Let the Hidden Markov Model(HMM) with the states may be Y, at initial stage probabilities p I of being in state i and transition probabilities a of transitioning from state i to state j. Say we observe outputs . The state sequence most likely to have produced the observations is given by the recurrence relations.

Vok= P (Xo/k).k

Vt,k = P (xi,j).pk /k).maxy∈Y(ay,kVt-1,y

Here Vt,k is the probability of the most probable state sequence responsible for the first t +1 observations (we add one because indexing started at 0) that has k as its final state. The Turbo path can be retrieved by saving back pointers which remember which state y was used in the second equation. Let Ptr (k,t) be the function that returns the value of y used to compute Vt,k if t > 0, or k if t = 0. Then:

yt = arg maxy∈Y(VT,y)

**Hidden Markov model and Turbo decoder**

**Hidden Markova model**

The chain of Markov is generally absorbed in noise processing signals. Markov chain is symbolized as {Xk}k≥0, hear k is basically an integer index. So as to quit the finite set that is for making secreted, Markov chain is hidden and can’t be observed in arbitrary state, thus it is experimental known to be as stochastic process {Yk}k≥0 this is an another linked process, as Yk is governed with the Markov chain in the distribution links [14]. This hidden Markova model is known to be a bivariate discrete time process {Xk , Yk} k≥0, where {Xk},{Yk} are the sequence of random independent variables as {Xk} is the Markov chain and conditional distribution of Yk. The hidden Markov model (HMM) is a signal facilitates to communicate with speech signals which achieved acceptance from almost all the communication systems.

The fully discrete model with an idea of conditional independence had introduced the hidden Markov modes as a bivariate process. The hidden Markov models consist of two classic layers sub cellular location known as upper layer and the functional class, which is lower layer. If any process is undertaken in the hidden Markov model the doubly stochastic process can’t be observed directly since, it is hidden and may be observed only with another stochastic process which will facilitates in sequential observation

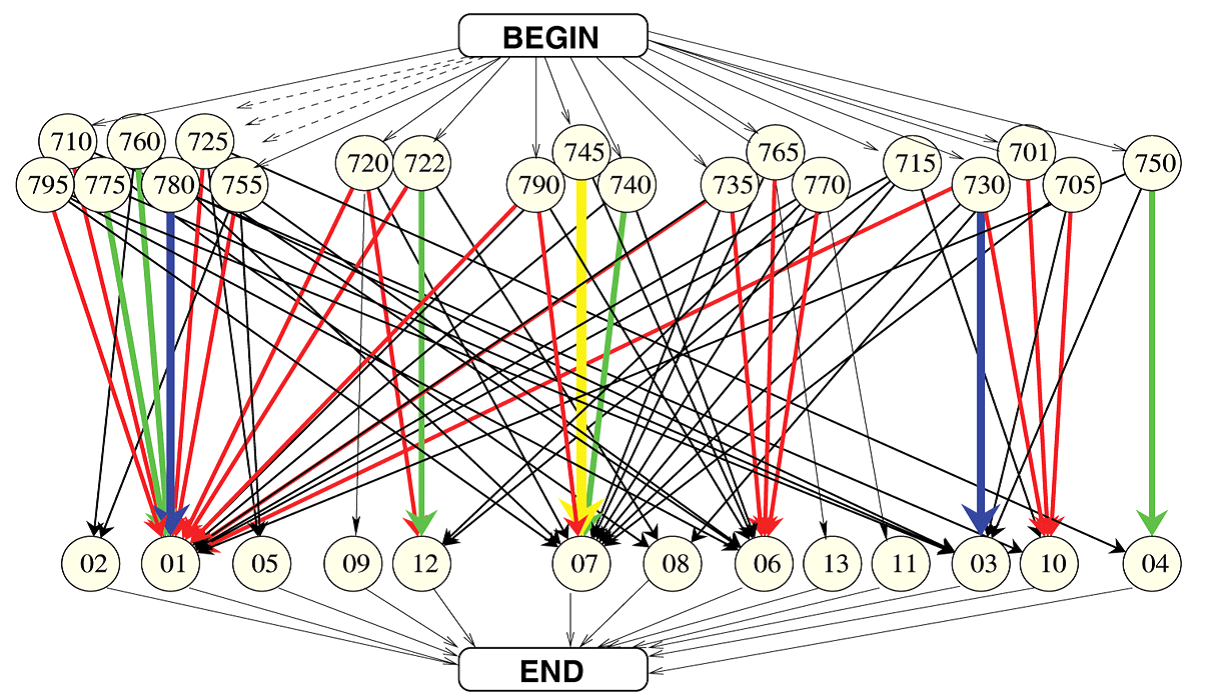


Figure 4.1.: Shows the hidden Markov model

Multiple paths of flow from start to finish can be examined using the upper and lower layers of the two-layer stack. These nodes at the endpoints of two layers encode the standards that are arbitrarily hidden from the top layer, notably location class variables, with the bottom that is functional class variables. The direction of the arrows in between two layers is a transition flow indication, and the hues and tints are displayed as per the predicted probability counts based on training sequences.

**Turbo decoder:**

An FEC-based forward error correction (FEC) technique is used in most Turbo decoders to encode and decode fragments of data. Decoding convolutional information is the primary use of the turbo decoder since it is capable of handling the large number of mistakes that are generated by channel noise. Decoding convolutional binary codes (viewed as a trellis tree) is made easier with the Turbo decoding algorithm, which is at the cutting edge of current communication standards (such as Qualcomm's TURBO standard). This Turbo decoder is utilised in the implementation of input code symbol stream to decode with a predetermined sequence. There are three key processing steps in the Turbo algorithm, which are outlined below:

• Branch metric generation

• State metric generation

• Chain back

There must be a precise order in which the noise and Markov processes are collected and correlated before performing the Turbo algorithm. The MLSD and MAP sequence detectors are used to drive the turbo detector's ISI channels, which have the predefined memory noise applied to them. The following are a few of the most essential aspects of Turbo decoder:

In most of the Industry standard k = 7. Where (G0, G1) = (133, 171), rated at ½ Turbo decoder. It is possible to implement both with Xilinx FPGA or ASIC. There are 256 latency clock cycle, Speed of the design is very high which is approximately up to 122 Mbps for the Virtex II at the same time for Spartan III the data rate is nearly 108 Mbps and more high for ASIC. The software input is of almost 4 bits. The length of track back will be of 64. Simple clock designs are completely synchronous.

**Block Diagram of Turbo algorithm**

The Turbo algorithm is one of the standard sections in number of high-speed modems of the process for information infrastructure applicable in modern world. The dynamic algorithm includes some path metrics so as to compute the path sequence transmitted earlier the name Turbo algorithm arrived after Andrew Turbo and is represented as VA for reorganization, record of huge possibility decodes as well as least reserved decoding are generally similar in a defined binary symmetric channel. Kia, J. (2005, p.1) explains Turbo algorithm as a “dynamic algorithm that uses certain path metrics to compute the most likely path of a transmitted sequence” [13]. The basic performance of the Turbo decoder is analyzed with the block diagram shown below. It consists of three main blocks branch metric unit, add compare select and trace back unit. The unit of branch metric will calculate all the branch metrics and then processed to add compare for selecting the surviving branches as per the branch metrics finally the decoded data bits are generated by the trace back unit.

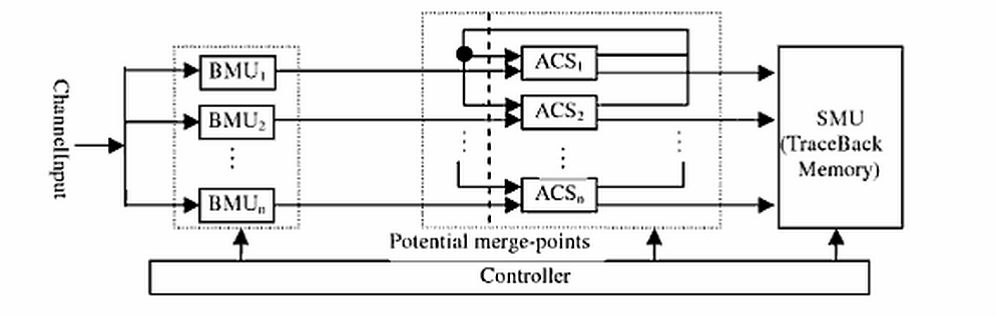


Figure – 4.2: Shows the basic block diagram of Turbo decoder [14].

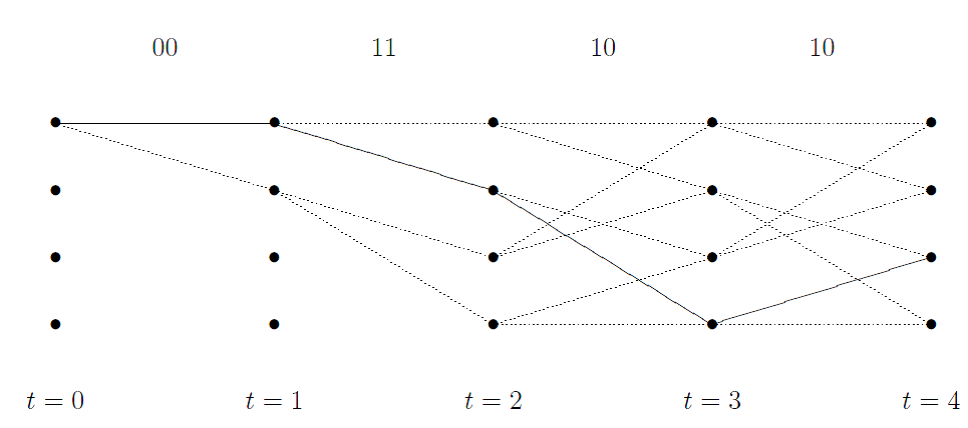


Figure 4.3: Shows the Turbo algorithm trellis [15].

For calculating the branch metric can be obtained with the trellis using the Euclidean analysis as follows:

BM (rr, bb) = (r0-b0)2+ (r1+ b1)2

= r02−2r0b0+b02+r12−2r1b1+b12

= r0b0+r1b1

Where,

rr = symbol received at the input

bb = branch symbol

Both rr and bb are dependent on the used for conventional encoder. Under the basic assumption that there is no noise in the data and the value of r and b will vary between -1 to +1, the range of branch metric will range within -2 to +2. In case rr = bb branch metric would be 2, Similarly r0 = - b0 as well as r1 = - b1 and BM= -2

The path metric (λ) in the minimum Euclidean distance in the trellis does not required the actual value the original order of the floating point pair numbers is

λnew=λprev+r0b0+r1b1

The path metric λ is the shortest distance among cumulative state, thus distance of the path (Euclidean distance) is inversely proportional to the branch metric. After complication of generating a trellis it is necessary to find survivor path with maximum path metric. In the above the solid black line is the survivor path.

**Description of Turbo Algorithm**

Turbo algorithm is basically implemented to decode the errors found in convolution encoded sequence. As discussed the Turbo algorithm will make use of trellis structure in finding the coded sequence based on the transmission signals. Since each and every code sequence will follow based on the trellis process of encoding data. Considering an example of trellis diagram of half rate, three convolution encoder K=3 and 15 bit messages with four possible states shown in 4 horizontal rows with dotes.

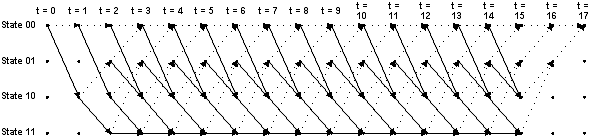


Figure: 4.4.shows the trellis diagram of turbo algorithm.

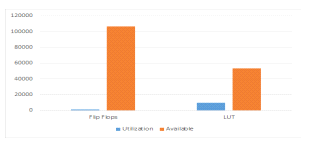


Fig. 4.5: The logic utilization of the Turbo encoder with parallel computation.

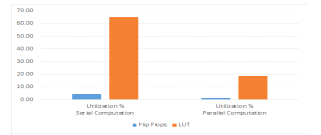


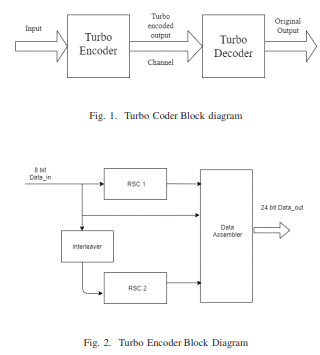
Fig. 4.6: The logic utilization of the Turbo encoder, serial vs parallel computation.

**5. PROPOSED METHOD**

Source system faults can occur when data is transmitted from the source system to the destination system. For this, the original message must be corrected. Channel coding techniques took a quantum leap forward with the introduction of turbo codes in 1993, and this was a watershed moment for modern digital communications. A sophisticated error-correcting code known as Turbo codes is currently available. ' An iterative decoding strategy that depends entirely on simple component code to reach close channel capacity has inspired the coding community in Turbo codes. Turbo encoders and turbo decoders make up the architecture of a turbo coder (Fig 1). There are two RSCs in the encoder, as well as an interleaver. For better random code performance, this article employs pseudo-random interleavers. These interleaved versions of the code tend to be long and scrambled. RSC encoders are used instead of typical convolutional encoders in turbo code implementation because they generate low weight parity codes. For turbo encoded data, MAP is used to decode and verify an error-free decoded data following the decoding process.

**6. IMPLEMENTATION**

A. Architecture of Turbo Coder a turbo encoder and decoder form the backbone of a turbo coder (shown in figure 1). There are two identical RSCs and one pseudorandom interleaver that make up the Turbo Encoder (figure 2). Parallel concatenated turbo codes are used in LTE to transmit data at a third of the pace. Each RSC works with a different set of information for each of its tasks. The first encoder receives the original data, while the second encoder receives the interleaved version. Interleaving is a technique that scrambles data bits using a specific algorithm. The interleaving algorithm has a significant impact on a decoder's performance when used. Using the RSC1 and RSC2 encoder outputs and systematic input, a 24 bit output is formed, as shown in Figure 6, from the turbo encoder output. Turbo decoder will get this across the channel. There are two SISO decoder modules with two pseudorandom interleavers and a pseudorandom deinterleaver in Figure 3 of a typical turbo-decoder block diagram.



The BCJR algorithm is the most used method for deciphering turbo codes. Iteration between two SISO part decoders is the essential and basic notion behind the turbo decoding method depicted in figure 3. To improve and revise the estimation of the original information bits, a pair of decoders work simultaneously. The convolutional code created by the first or second CE is decoded by the first and second SISO decoders, respectively. Turbo-iteration refers to the process of performing one pass through the first component decoder followed by another pass through the second component decoder in rapid succession.

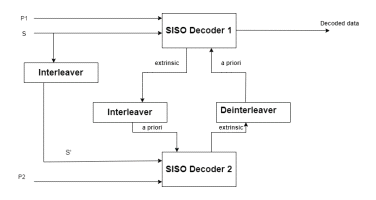


Fig. 3. Turbo Decoder Block diagram

**B. SISO Decoder**

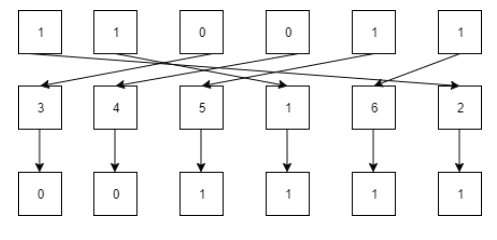
The soft-in-soft-out (SISO) decoder receives the true (soft) value of the signal at its input. Estimates of the number of bits in each input For each bit of data received, the decoder calculates a probabilistic approximation of whether it is one or zero. For the SISO component decoder, the turbo-decoder under discussion in this study employs the MAP algorithm. A valid path through the trellis is not a constraint on the MAP algorithm's set of bit estimations. Therefore, the outputs of a Viterbi decoder that detects the most probable path across the trellis should differ from those obtained by that decoder. 1) The MAP Algorithm: In order to choose the most likely bit at each trellis point, the MAP algorithm uses the complete sequence that was obtained. An N-bit encoded symbol frame and the decoder's output y are shown in this example. A MAP decoder generates 2m a posteriori probabilities for each dsym i. Making a difficult choice about the value j that is equal to dsym I aids in increasing the a posteriori probabilities to the possible extent.

It is expressed injoint probabilities as: P r(dsym i = j|y) = P(dsym i = j, y) 2m−1 k=0 (P(dsym i = k, y) (1)

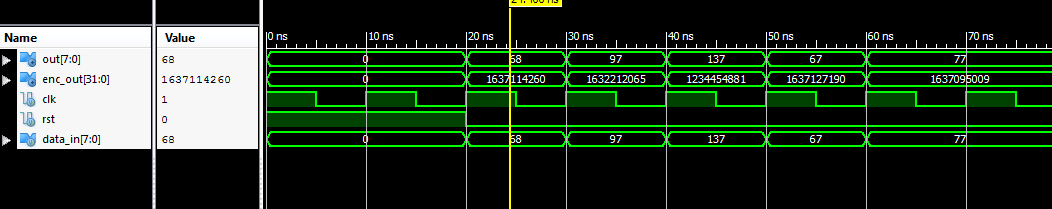
The trellis form of the code makes it possible to decompose the joint probability of the previous and subsequent data. In Equation 2, the forward recursion metric i(S) is shown. It instantaneously provides the probabilities of state S based on prior channel values at i. It is also used to calculate the probability of the state based on the upcoming values of the channel and Branch metric (S,S) backward-recursion metric. P r(dsym I = j|y) = (2) As for the branch metric, p(yi|xi) = i(S &amp; S) A channel transition probability of p(yi|xi) is defined as follows: p(yi|xi) = p(yi|xi) where xi = ith sent modulated signal and yi = ith received signal. The a priori probability is 1/2m for an equiprobable source. Equation 1 eliminates the symbol channel input by adjusting the branch metric.

**C. Interleaver**

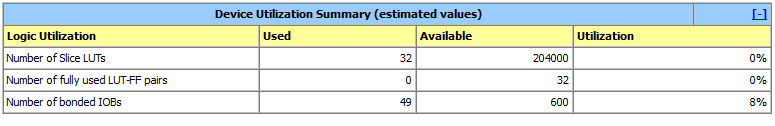
Designing interleaver for the turbo code is critical to its performance Convolutional encoders can't tell the difference between neighbouring bits if interleavers scramble the data in a pseudorandom order. The interleaver is utilised on both the encoder and the decoder. It produces a large block of data on the encoder side, while it compares two SISO decoders' outputs and fixes the fault on the decoder portion. Pseudo-random deinterleaver works in conjunction with pseudo-random interleaver to provide an additional layer of security.



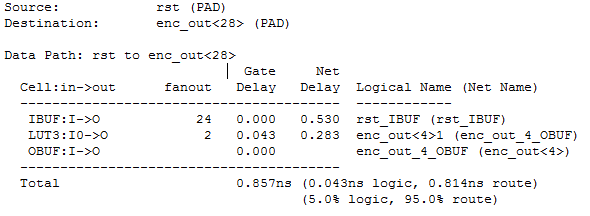
**7. SIMULATION RESULTS**



**Simulation outcome**



**Design summary**



**Time summary**

**Total 0.857ns (0.043ns logic, 0.814ns route)**

**(5.0% logic, route)**

**8. CONCLUSION**

The Turbo method is considered because it is more intriguing and harder for this research issue and has a wide range of applications in the digital communications industry. As a result of this study, software developers who use the Turbo algorithm stand to gain an advantage in the market. Non-students can analyse the Turbo algorithm concepts and learn more about it. In this study, VHDL coding is used to implement the Turbo algorithm. The primary benefit of the Turbo algorithm is error correction using VHDL, which consumes less power. Anyone who reads this text will have to learn how to use various tools, such as Xilinx ISE and MODELSIM, to be effective.

As data must be transferred from one location to another, the likelihood of encountering errors is higher because of the extensive coding required. Using some of the main elements of the Turbo algorithm, random problems can be solved by interrupting the original bit sequence and using some of the important features of the Turbo algorithm. forward error correction (FEC), auto repeat request (ARQ), hybrid ARQ, and error code correction (ECC) are some of the most common methods for correcting errors in data (channel coding). MATLAB and C can be utilised to implement Turbo algorithm. Isolated sign language Turbo algorithm and continuous sign language Turbo algorithm are both utilised to search the frame simultaneously. The Turbo algorithm, a dynamic programming algorithm, is used to determine the series of hidden states known as the Turbo path. The MATLAB code used to evaluate the Turbo algorithm should be as minimal as possible. Assembler code for the Turbo decoding algorithm is written in TMS320C54x

In the world of high-speed integrated circuits Very High-Speed Integrated Circuit (VHSIC). Using this lingo, we can describe the behaviour of field-programmable gate arrays. VHDL serves as a general-purpose computer programming language. VHDL is similar to C and C++ languages. The primary purpose of VHDL is to specify a circuit's function. Using VHDL, it is possible to construct a test bench to verify the functioning of the plan by defining stimuli and comparing them to the user's results. As a result, the description will be lower even in the presence of more faults, and the algorithm will perform more effectively. A further benefit of utilizing this turbo method comes from its low cost of implementation.

MODELSIM – Simulation and XILINX-ISE are the instruments utilised in conjunction with it for the implant. ModelSim SE and ModelSim DE are the two most basic commercial products. This software environment is known as the Xilinx Integrated Software Environment, or ISE. Xilinx ISE is the most often used software for HDL design and development. Turbo decoder gate level simulators and high-speed applications are employed in these decoders in order to save electricity. The Turbo decoder is concerned with numerous processing elements in order to achieve a correct state at any given time. The radix butterflies employ two register files, one for writing and one for reading. The path metrics file is used to hold the bits. Multipath fading adaptive technique is utilised. Turbo decoder techniques can be used to enhance the throughput of the encoded data. Finally, the Turbo algorithm has been successfully implemented utilising Verilog HDL hardware and tools like Xilinx and FPGA. All of the code's workings, including its design and synthesis results, may be easily found using Xilinx ISE and FPGA editors. The Adaptive Turbo algorithm necessitates the development of a less memory-intensive Turbo algorithms architecture since it demands considerably increased memory locations and the ability to do logical programming operations. It is necessary to select the algorithm parameters and also the essential noise level fluctuations in order to get better results.

The Turbo encoder module is built inside the IVS modem as an embedded component. The Turbo encoder module was created using FPGA technology. For the design and simulation of the module, Verilog HDL and Xilinx tools are utilised. The encoding procedure makes use of both serial and parallel computing methods. A smaller chip and faster module processing time can be achieved through parallel computation. By using parallel computation encoding, the processing time is reduced by 58 percent and the logic consumption by 73 percent over the serial computation technique. Both simulation and chip processing analysis show an increase in processing speed.

**Recommendations**

The Turbo algorithm is advised to overcome the problem of power supply in applications that require high decoding throughput in order to achieve an output of several hundred Mega Bits per second. Relaxed Turbo, a novel algorithm, can help reduce silicon area and power consumption even more, making it possible to save even more power and silicon area. A Turbo algorithm that uses less memory is recommended because the Adaptive Turbo method necessitates an enormous quantity of logic and memory to complete its tasks. In the development of the implementation, FPGA kits are employed. The FPGA, which is employed by Adaptive computing, will spend a significant amount of time in milliseconds overwriting the data and consuming more power to charge the assembled data. In the communication path, FPGA can be fatal and guide transient growth in response time [42].

In order to circumvent these dynamically reconfigurable devices, dynamically reconfigurable processors are employed. The difficulty of localization principle shows a low signal-to-noise ration. 3-D To get around this, we suggest using a turbo search algorithm. A more advanced version of the method is highly recommended because of its great throughput.

**Future scope**

Future decoding improvements will be possible with the help of FPGA and hybrid microprocessor technology. Sequential decoding provides additional power gains. The adaptive array approach will be utilised in future satellite communications to mitigate signal degradation caused by multipath fading. Noise level and algorithmic factors determine the adaptive turbo decoding's solutions. There is a fixed complexity and noise level independent M-algorithm utilised. To boost decoder performance, the adaptive Turbo algorithm will be implemented on reconfigurable hardware in the future. [43] The power saving architecture can be created for the aforesaid decoder, which can be run on mobile devices, in order to save power. The Turbo decoder may be able to decode non-binary codes in the future. XILINX has created a turbo decoder, but it can also be implemented using JAVA in the future. The Turbo algorithm can therefore be employed in a variety of future circumstances. Because of this, it is possible to substantially reduce the complexity in the future. A significant reduction in decoding noise can be achieved by utilising the M algorithm.

**Bibliography:**

* [1] R. E. Stake (1995) “The art of case study research”. USA: Sage.
* [2] K.Aleksandar, M.F. Jose (2000) “The Turbo Algorithm and Markov Noise Memory”. [Internet] available at URL:
* <<http://www.ece.cmu.edu/~moura/papers/kavcic_turbonoisememory.pdf>>, [accessed on20th November 2010].
* [3] K. Hueske, J. Geldmacher, J. Gotze (2007) Adaptive decoding of convolutional codes, Copernicus Publications, pp. 209-214. Internet. (N.D) Coding in communication system.[Internet] available at URL: <<http://www.mathdb.org/notes_download/elementary/number/ne_N6.pdf>>, [accessed on20th November 2010].08)e: Sprnger.
* [4] B. Tristan (2006) IMPLEMENTATION OF THE TURBO ALGORITHM USING FUNCTIONAL PROGRAMMING LANGUAGES. [Internet] available at URL: <<http://people.eecs.ku.edu/~esp/publications/c2009ItcTristan.pdf>>, [accessed on 20November 2010].
* [5] K. Hucker (2001) Research Methods in Health, Care and Early Years. UK: Heinemann Publishers.
* [6] Y. K. P.Cheung, G. A. Constantinides, J.T. D. Sousa (2003) Field-programmable logic and applications: 13th International Conference, FPL 2003, Lisbon, Portugal. New York: Springer.
* [7] J. Kia (2005) What Is a Convolutional Encoder / Convolutional Encoding. [Internet] available at URL: <<http://www.askkia.com/articles/what-is-a-convolutional-encoderturbo-algorithm.html>>, [accessed on 20 th November 2010].
* [1] M. Viterbi, “Shannon theory, concatenated codes and turbo coding”, <http://occs.donvblack.com/viterbi/index.htm>, 1998.
* [2] Shannon CE. “Communication in the presence of Noise”. Proceedings IRE, Jan 1949, reprinted Proceedings IEEE, Vol. 86, and no. 2 Feb 1998.
* [3] J. B. Cain, G. C. Clark, Jr., and J. M. Geist, “Punctured Convolutional Codes of Rate (n¡1)=n and Simplified Maximum Likelihood Decoding,” IEEE Transactions on Information Theory, vol. IT-25, pp. 97–100, Jan 1979. <https://doi.org/10.1109/TIT.1979.1055999>.
* [4] C. Berrou and A. Glavieux, “Near Optimum Error Correcting Coding and Decoding: Turbo-Codes,” IEEE Transactions on Communications, vol. 44, pp. 1261–1271, Oct. 1996. <https://doi.org/10.1109/26.539767>.
* [5] Oscar Y. Takeshita and Daniel J. Costello Jr,“New deterministic interleaver designs for turbo codes”, IEEE Transactions on Information Theory, Vol. 46, Issue 6, pages 1988–2006, Sept. 2000. <https://doi.org/10.1109/18.868474>.
* [6] Arya Mazumdar, A. K. Chaturvedi, and Adrish Banerjee, “Construction of turbo code interleavers from 3-regular Hamiltonian graphs”, arXiv:cs.IT/0512093 v1, December 2005.
* [7] Libero Dinoi and Sergio Benedetto, “Design of fast-prunable srandom interleavers”, IEEE Transactions on Wireless Communications, Vol. 4m No. 5, pages 2540–2548, September 2005.
* [8] D. Divsalar and F. Pollara, “On the design of turbo codes”, TDA Progress Report 42-123, November 1995.
* [9] Jonghoon Ryo and Oscar Y. Takeshita, “On quadratic inverses for quadratic permutation polynomials over integer rings”, IEEE Transactions on Information Theory, Vol. 52, No. 3, pages 1254– 1260, March 2006. <https://doi.org/10.1109/TIT.2005.864442>.
* [10] Stephane Y. Le Goff and Faisal Obaid Al-Ayyan, “On the design of bit-interleaved turbo-coded modulation”, IEEE Proceedings Information Theory Workshop, pages 73 – 75, September 2001.
* [11] EirikRosnes and OyvindYtrehus, “On the design of bit interleaved turbo-coded modulation with low error floors”, IEEE Transactions on Communications, Vol. 54, No. 9, pages 1563–1573, September 2006. <https://doi.org/10.1109/TCOMM.2006.881215>.
* [12] Stephane Y. Le Goff and Faisal Obaid Al-Ayyan, “Design of bit interleaved turbo-coded modulations”, IEEE Electronics Letters, Vol. 37, No. 16, pages 1030–1031, August 2001. <https://doi.org/10.1049/el:20010702>.
* [13] Adrish Banerjee, Francesca Vatta, BartoloScanavino, and Daniel J. Costello Jr., “Nonsystematic turbo codes”, IEEE Transactions on Communications. Vol. 53, No. 11, pages 1841–1849, November 2005. <https://doi.org/10.1109/TCOMM.2005.858672>.