Design of (2,1,4) Convolutional Encoder and Viterbi Decoder using Verilog

Archit Halder[1], Saurabh Gajanan Bhokare[2] and Dr.Basanta Bhowmik[3]

Department of Electronics & Communication Engineering

NIT Jamshedpur

E-mail:- *bb.cemk.ece@gmail.com[3]*

DOI: - https://doi.org/10.47531/MANTECH/ECC.2021.21

Abstract

The present study demonstrated the use of various error detection and correction methods that provide a dependable quality of data transmission. Present technological development demands efficient and reliable data transmission systems. During data transmission (from source to destination), the high possibility of erroneous data at the destination (due to the inclusion of noise) is a general phenomenon. Keeping in mind such obstacle present paper addressed one technique to expound the digital design conversion of the Viterbi decoder. The decoder efficaciously exploits the Viterbi algorithm as well as carries out digital design conversion of (2, 1, 4) Convolution Encoder with a constraint length of L=5 and code rate of ½. The design has been verified employing Verilog hardware description language (HDL). Further, synthesis followed by simulation was carried out using XILINX ISE 14.5 simulator.

Keywords: - *(2,1,4) Convolutional Encoder, Viterbi Decoder, Viterbi algorithm and Trellis structure.*

INTRODUCTION

From the beginning of the communication era, encoding and decoding become important technique for reliable communication [1-2]. This transmission and recovery of data have to be done without errors. Even with the advent of communication technology from telegram to digital communication, the most concerning issues for researchers and engineers are none other than the efficiency and reliability of communication channels [1-3]. There are various techniques to recover data, better the error correction better the technique. Improvement in the capacity of error correction can be made by incorporating redundant information for the source data transmission. In late 1940, Shannon, Golay, and Hamming devoted so much effort in the field of error detection and correction for digital communication [1-4]. Shannon defined the limits of reliable communication theoretically, whereas Golay and Hamming were developing various practical schemes for error detection and control. The codes such as Hamming code and Golay code were linear block code. Another approach of coding was introduced in 1955 called as convolutional coding [5].

In 1965, J.E. Savage presented various computational problems of sequential decoding. Savage indicated that overflow probability is a strong function of source rate, and to achieve small overflow probabilities, the source rate need not to be less than 90 % of the computational cut off rate [6]. Andrew Viterbi proposed an algorithm in 1967, used for decoding bit-stream encoded using FEC code [6]. The Forward Error Correction (FEC) techniques are used in system applications for the transmitter to encode and receiver to correct errors. In 2007, Hema. S, Suresh Babu. V and Ramesh. P presented an implementation of a Viterbi decoder whose constraint lengthand code rate are 11 and 1/3, respectively. The paper showed that the constraint length is proportional to the strength of the code. [7]. A combination of both the Convolutional Encoder and Viterbi Decoder are operated simultaneously to provide the error correction [8]. Original data, which is a continuous stream of redundant data, is provided by Convolutional Encoder The Viterbi Decoder, on the other hand, incorporates maximum decoding to recover the data in the presence of additive white Gaussian noise (AWGN). In 2013, Dhobale, R., Kalyani Ghate, Nikhil Pimpalgaonkar and R. B. Khule presented a low power option of Viterbi Decoder, but the design fails to handle

more complex application requirements [10]. In general, the Viterbi algorithm has many applications and works on the concept of forming a trellis structure. The communication system consists of an encoder, channel and decoder. The encoder is a convolutional encoder that encodes the message and generates the encoded bits, which are transmitted through the channel. At the receiver end, there is Viterbi Decoder which uses the Viterbi algorithm to decode the received message. As most of the Viterbi decoders are Parameterizable intellectual property (IP), they are expensive to use due to the patent issue. Thus to realize them on FPGA is demanding. The present paper explores the convolutional encoder and Viterbi decoder of the format (2,1,4).

CONVOLUTIONAL ENCODER

A convolutional code is a type of error correction code used as an alternative to the block codes. Convolutional codes are different from the block codes in many ways, such as there are no distinct sections or blocks to make up code words, but unwanted bits are added throughout the data. Another is that the convolutional encoder consists of memory elements that work on the concept of the Mealy machine. In the Mealy machine, the output is a function of the current state and current input given in Table I

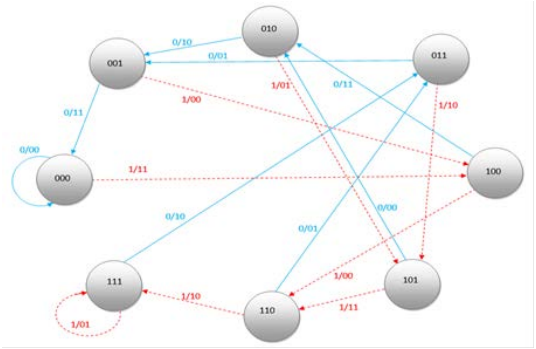

Fig. 1: State Diagram for (2,1,4) Convolutional Encoder

Table I: State table for (2,1,4) Convolutional Encoder

In Bits	Input State			Output States			Out Bit	
I_1	S_1	S_2	S_3	S_1	S_2	S_3	O_1	O_2
1	0	0	0	1	0	0	1	1
0	0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	0	0
0	0	0	1	0	0	0	1	1
1	0	1	0	1	0	1	0	1
0	0	1	0	0	0	1	1	0
1	0	1	1	1	0	1	1	0
0	0	1	1	0	0	1	0	1
1	1	0	0	1	1	0	0	0
0	1	0	0	0	1	0	1	1
1	1	0	1	1	1	0	1	1
0	1	0	1	0	1	0	0	0
1	1	1	0	1	1	1	1	0
0	1	1	0	0	1	1	0	1
1	1	1	1	1	1	1	0	1
0	1	1	1	0	1	1	1	0

Convolutional codes are also stated as trellis codes. Convolutional codes are defined using two factors viz; code rate and constraint length. A code rate (r) is represented by k/n, where 'k' is the number of bits given as input to the convolutional encoder and 'n' is the number of bits in the output of the encoder. Code rate measures efficiency. The lower the code rate higher is the efficiency. Constraint length (L) denotes the length of the encoder, also expressed as the number of shift registers (m) used in the encoder to feed the combinational logics such as adder to get the output without any feedback; L=k(m-1). The constraint length affects the performance of convolutional code. Larger the constraint length powerful the code with more gain but on the falling hand, more complexity and delay in decoder [8-9][11].

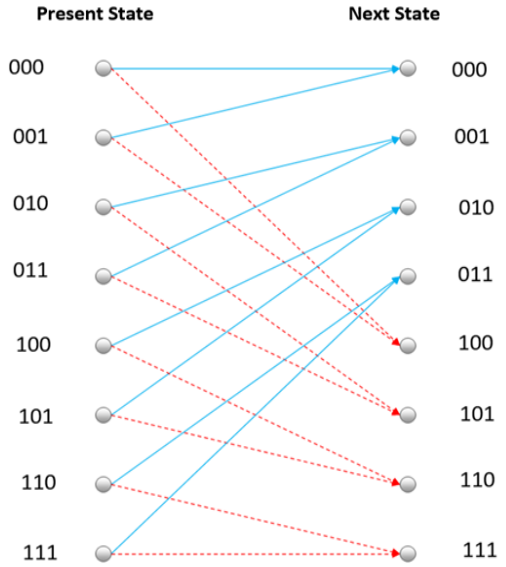

Fig. 2: Trellis Structure for (2, 1, 4) Convolutional Encoder

Table II: Parameters for (2, 1, 4) Convolutional Encoder

Definition	Symbol	Value
Input Bits	k	1
Output Bits	n	2
Code Rate	r = k/n	½
Constraint Length	L	5

The encoder is designed with a code rate (r) of ½, and the constraint length (L) of shift registers is 5. The information feed to the shift registers is sequential in nature of one bit at a time. Thus forming a (2, 1, 4) encoder. The decoder consists of shift registers and XOR gates. The output, taken from the XOR gate, is known as code symbols. In the initial, all the stages of the encoder are made zero. The generator polynomial equations determine the output code.

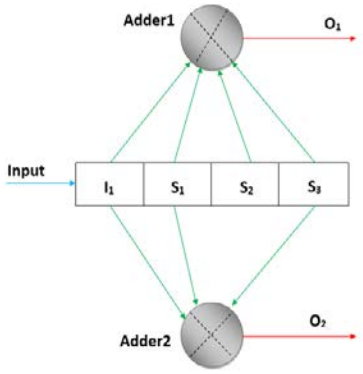

Fig. 3: Structure showing (2,1,4) Convolutional Encoder

TRELLIS STRUCTURE

The convolution encoded data is decoded using the trellis structure in the Viterbi algorithm [12]. A trellis state of the Viterbi decoder with ½ rate convolution encoder of the constraint length 5, the complete Trellis structure for the Viterbi Decoder is shown in Fig 4.

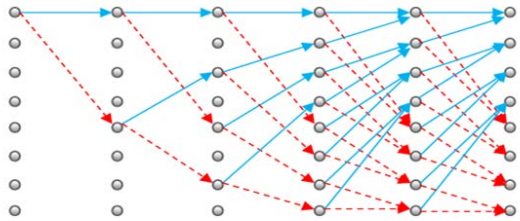

Fig. 4: Complete Trellis structure for Viterbi Decoder

The Trellis flow has been discussed in Fig 5-6. The comparison for the state metrics is shown for each state, followed by the decoding process for all the state transitions. The transition has done with the survivor memory unit, which makes decisions on the input bits by eliminating all the path leaving one. There is a starting point of the branches for the state metrics. There are a total of 2 branches emanating from state 000 to 000 and 100. Based on each branch, there is a specific input and output assigned to each branch. Likewise, for the other states, this process continued as reveals in Fig 5 (where the survivors with hamming distances are shown) until the last state metric reached. The last remaining path has been considered for decoding the output bits and corresponding message input bits (shown in Fig 6).

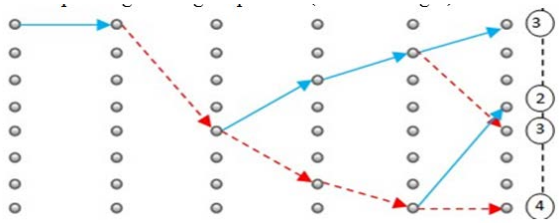

Fig. 5: Survivors with hamming values

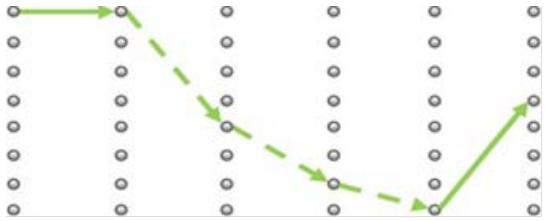

Fig. 6: Desired path for the Viterbi Decoder

VITERBI DECODER

The Viterbi decoder uses the concept of the Viterbi algorithm to obtain output from the received signal code. The Viterbi algorithm is brought into reality using hardware blocks such as Branch Metric Unit (BMU), Add Compare Select (ACS), Survivor Memory Input (SMI), and Path Metric Unit (PMU) [9]. Here the encoded input is given to the BMU, which gives the branch metrics.

The BMU is connected to ACS, which forms a loop together with PMU and further to SMI. The values of the Branch Metrics are fed to the ACS unit in a recurrent fashion, whereas the Path Metrics values are being compared in the PM unit. The final values incorporate all the above blocks and especially with the traceback module, and generates the decoded value.

Branch Metric Unit (BMU) is used to calculate branch metrics, i.e. the separation of hamming distances from the received codeword to the estimated output codeword. Hamming distance is the other name for Branch Metrics. For example, if the received code is 11, then the hamming distance with 10, 01, 11 and 00 is 01, 01, 00 and 10, respectively.

Add Compare Select (ACS) unit adds the input data from BMU and PMU and compare them to obtain the current state metric and next state metric of the loaded data[9]. The decision bit generated is forwarded to SMU. The SMU stores the decision bit generated by the add compare select unit and uses them to compute the decoded output.

IMPLEMENTATION

Verilog HDL has been used to implement the (2, 1, 4) Convolutional Encoder and the corresponding Viterbi decoder, moreover full-custom design has been taken into consideration while developing the Verilog code. The (2, 1, 4) Convolutional Encoder module implements an FSM which effectively computes the output bits and output state based on the provided set of inputs. This particular design is based on XOR gate and shift registers.

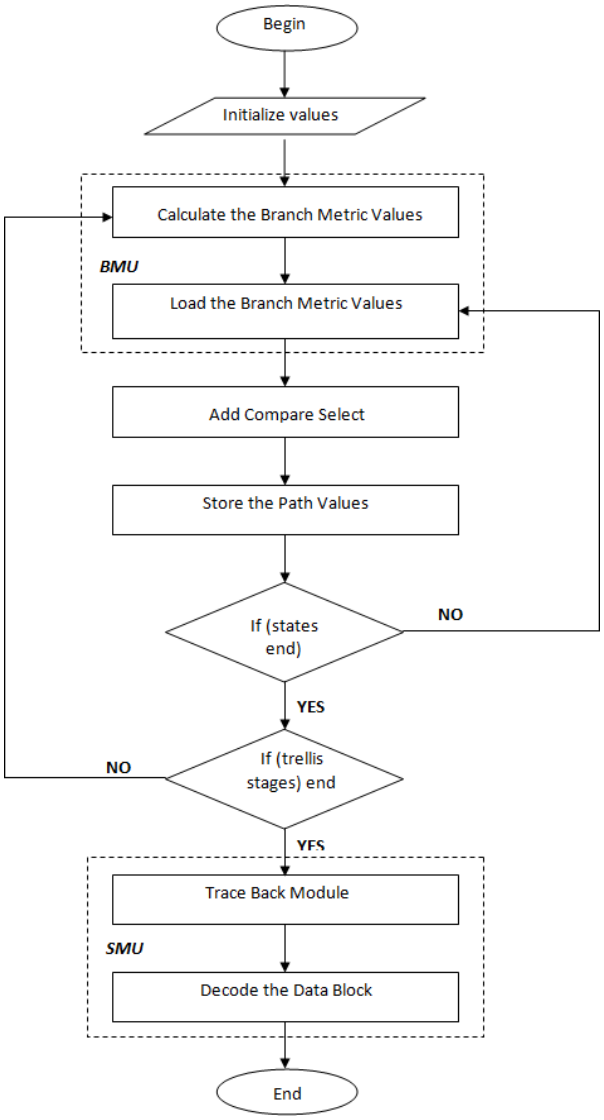

Fig. 7: Design Flow of Viterbi Decoder

For decoding purpose, the Viterbi algorithm has been used in the Viterbi decoder. The design flow of the Viterbi decoder is shown in Figure 7.

As shown in the design flow, first, we need to load the Branch Metric Values from the trellis structure [13]. After that, the loaded values are then augmented into ACS (Add Compare Select) Unit, where the values are first added to the partial path metrics to obtain a new path metric. After that, a comparison was taken out when the two paths arrive at the same state.

After the comparison, the path having the least metric will be chosen; this particular path is known as the survivor path. This process is done for all the states in the trellis diagram. Lastly, after finishing with all the states, the final survivor path is fed into a traceback module, which produces the decoded data (original input data) [14-17].

RESULTS AND DISCUSSIONS

```
=========================================================================
*                        HDL Synthesis                                  *
=========================================================================
Synthesizing Unit .
    Related source file is "C:\Users\trans\Desktop\Verilog\work\Papework\
        s0 = 3'b000
        s1 = 3'b001
        s2 = 3'b010
        s3 = 3'b011
        s4 = 3'b100
        s5 = 3'b101
        s6 = 3'b110
        s7 = 3'b111
    Found 2-bit register for signal .
    Found 3-bit register for signal .
    Found finite state machine  for signal .
    -----------------------------------------------------------------------
    | States             | 8
    | Transitions        | 16
    | Inputs             | 1
    | Outputs            | 3
    | Clock              | clk (rising_edge)
    | Reset              | reset (positive)
    | Reset type         | asynchronous
    | Reset State        | 000
    | Encoding           | auto
    | Implementation     | LUT
    -----------------------------------------------------------------------
    Summary:
        inferred   2 D-type flip-flop(s).
        inferred   7 Multiplexer(s).
        inferred   1 Finite State Machine(s).
Unit  synthesized.
=========================================================================
```

Fig 8: Synthesis report of (2,1,4) Convolutional Encoder

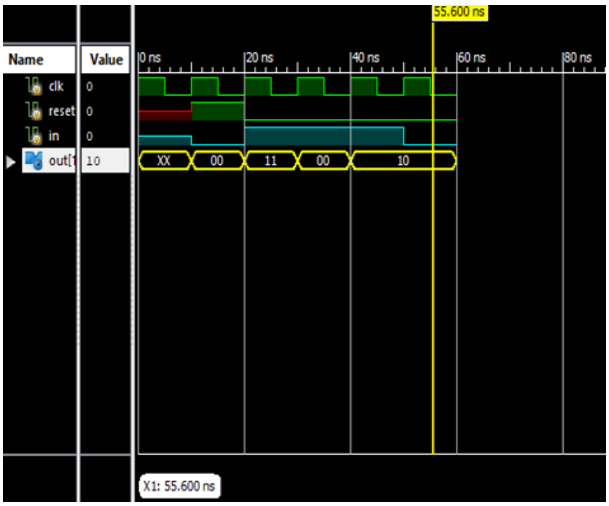

Fig 9: Simulation result of (2,1,4) Convolutional Encoder

The simulation of (2, 1, 4) Convolutional Encoder of L = 5 and r = 1/2 has been carried out using Verilog HDL and the simulation output and HDL synthesis reported has been generated. Input Bits and Encoded output bits are indicated as:

Input Bits X = [01110],

Encoded Output Bits Y = [0011001010].

The simulation of corresponding Viterbi decoder has been carried out and HDL report has been generated for the design implementation. Encoded Inputs, Decoded Outputs and Decoded Inputs are indicated as:

Encoded Input Bits Y = [0011001010],

Decoded Output Bits Z = [0001011010],

Decoded Input Bits M = [01110]

```
    Summary:
        inferred  90 Adder/Subtractor(s).
        inferred 369 D-type flip-flop(s).
        inferred  71 Comparator(s).
        inferred  46 Multiplexer(s).
Unit  synthesized.
=========================================================================
HDL Synthesis Report
Macro Statistics
# Adders/Subtractors                                   : 90
 32-bit adder                                          : 90
# Registers                                            : 80
 10-bit register                                       : 9
 3-bit register                                        : 46
 4-bit register                                        : 8
 5-bit register                                        : 9
 8-bit register                                        : 8
# Comparators                                          : 71
 3-bit comparator greater                              : 16
 3-bit comparator lessequal                            : 55
# Multiplexers                                         : 46
 10-bit 2-to-1 multiplexer                             : 15
 3-bit 2-to-1 multiplexer                              : 16
 5-bit 2-to-1 multiplexer                              : 15
# Xors                                                 : 20
 32-bit xor2                                           : 20
=========================================================================
```

Fig. 10: Simulation result of Viterbi Decoder

Fig. 11: Simulation result of Viterbi Decoder

CONCLUSION

In the present research endeavor, (2, 1, 4) Convolutional encoder of L = 5 and r = ½ and its corresponding Viterbi Decoder has been proposed and implemented employing Verilog HDL. This design of convolutional encoder proves to be a better version than it's predecessor i.e, (2,1,3) encoder in terms of BER. The Simulation and Synthesis has been carried out using XILINX-ISE Design Suite 14.5 The target device of the proposed design is Artix 7 based XC7A100T FPGA device. The encoding and decoding process has been performed by (2, 1, 4) Convolutional encoder and Viterbi decoder, respectively. The value of L can be extended >5 to reduce the probability of error for a given amount of noise but at the cost of more complexity in the design.

REFERENCES

1. E. Shannon, "A mathematical theory of communication," Bell Sys. Tech. J., vol. 27, pp. 379–423, 1948.
2. R. W. Hamming, "Error detecting and correcting codes," Bell Sys. Tech.J., vol. 29, pp. 147–160, 1950.
3. M. J. E. Golay, "Notes on digital coding," IEEE, vol. 37, p. 657, 1949.
4. J. Wesdock, and C. Patel, "Formulation of modulation recommendations for future nasa space communications," in IEEE Aerospace Conference, Big Sky, Montana, March 2008.
5. J. E. Savage, "Sequential decoding-The computation problem," Bell Sys. Tech. J., vol. 45, pp. 149-175, 1966.
6. A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," IEEE Trans. On Information Theory, Vol. 13, pp. 260-269, 1967.
7. Hema, S., V, S. B., & P, R. FPGA Implementation of Viterbi Decoder. 6th WSEAS International Conference on Electronics, Hardware, Wireless and Optical Communications, (Jan. 2007), 162–167.
8. Wong, Y.S. et.al "Implementation of convolutional encoder and Viterbi decoder using VHDL" IEEE Tran. on Inform. Theory, Pp. 22-25, Nov.2009.
9. G. S. Suganya and G. Kavya, "RTL Design and VLSI Implementation of an efficient Convolutional Encoder and Adaptive Viterbi Decoder", IEEE International conference on Communication and Signal Processing, April 3-5, 2013.
10. Dhobale, R., Kalyani Ghate, Nikhil Pimpalgaonkar and R. B. Khule. "Implementation of Adaptive Viterbi Decoder for Wireless Communication." International Journal of Science and Research (IJSR), India Online ISSN: 2319-7064 Vol. 2 Issue-3, March 2013.
11. Fazal Noorbasha, M. Manasa, R. Tulasi Gouthami, S. Sruthi, D. Hari Priya, N. Prashanth, And Md. Zia Ur Rahman, "FPGA Implementation of Cryptographic Systems For Symmetric Encryption", Journal of Theoretical and Applied Information Technology, Vol. 95. No 9, pp. 2038-2045, 2017
12. Akash Thakur and Manju K Chattopadhyay, "Design Implementation of Viterbi Decoder Using VHDL" 2018 IOP Conf. Ser.: Mater. Sci. Eng. 331, 01200, 2018
13. 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" IEEE Transaction on Inform Theory, vol. pp. 2011.

14. H. L. Kumawat and S. Sharma, "An Implementation of a Forward Error Correction Technique using Convolution Encoding with Viterbi Decoding," vol. no. 5, pp. 95–99, 2012.
15. T. Problem, "L ECTURE 9 Viterbi Decoding of Convolutional Codes,"Spring, vol. 2010, pp. 1–9, 2010.
16. "A Viterbi Decoder Using System C for Area Efficient," Comp. A J. Comp. Educ., no. September, 2006.
17. Xilinx Viterbi Decoder V 9.0 product guide 2014