RESEARCH ARTICLE OPEN ACCESS

A Low Power Viterbi Decoder for Trellis Coded Modulation System

M. Jansi Rani[1], S.Vidheswari[2]
[1, 2] Vivekanandha college of engineering for women, Tiruchengode
[1]M. Jansi Rani, M.E.,
[2]S.Vidheswari, PG-scholar

Abstract
Forward Error Correction (FEC) schemes are an essential component of wireless communication systems. Convolutional codes are employed to implement FEC but the complexity of corresponding decoders increases exponentially according to the constraint length. Present wireless standards such as Third generation (3G) systems, GSM, 802.11A, 802.16 utilize some configuration of convolutional coding. Convolutional encoding with Viterbi decoding is a powerful method for forward error correction. Viterbi algorithm is the most extensively employed decoding algorithm for convolutional codes. The main aim of this project is to design FPGA based Viterbi algorithm which encrypts / decrypts the data. In this project the encryption / decryption algorithm is designed and programmed in to the FPGA.
Keywords **-** Viterbi Decoder, Convolutional Encoder, Forward Error Correction technique.

I. INTRODUCTION

In telecommunication and information theory, forward error correction (FEC) (also called channel coding[J]) is a system of error control for data transmission, whereby the sender adds systematically generated redundant data to its messages, also known as an error-correcting code (ECC). The carefully designed redundancy allows the receiver to detect and correct a limited number of errors occurring anywhere in the message without the need to ask the sender for additional data. FEC gives the receiver an ability to correct errors without needing a reverse channel to request retransmission of data, but this advantage is at the cost of a fixed higher forward channel bandwidth. FEC is therefore applied in situations where retransmissions are relatively costly, or impossible such as when broadcasting to multiple receivers. The process of adding this redundant information is known as channel coding. Convolutional coding and block coding are the two major forms of channel coding. Convolutional codes operate on serial data, one or a few bits at a time. Block codes operate on relatively large (typically, up to a couple of hundred bytes) message blocks. There are a variety of useful convolutional and block codes, and a variety of algorithms for decoding the received coded information sequences to recover the original data.

Convolutional codes are employed to implement FEC. In telecommunication, a convolutional code is a type of error-correcting code in which

- Each *m*-bit information symbol (each *m*-bit string) to be encoded is transformed into an *n*-bit symbol, where *m*/*n* is the code *rate* ($n \geq m$) and
- The transformation is a function of the last *k* information symbols, where *k* is the constraint length of the code.

Convolutional codes are used extensively in numerous applications in order to achieve reliable data transfer, including digital video, radio, mobile communication, and satellite communication.

Convolutional codes work on bit or symbol streams of arbitrary length. They are most often decoded with the Viterbi algorithm, though other algorithms are sometimes used. Viterbi decoding allows asymptotically optimal decoding efficiency with increasing constraint length of the convolutional code, but at the expense of exponentially increasing complexity.

Several algorithms exist for decoding convolutional codes. For relatively small values of *k*, the Viterbi algorithm is universally used.

A Viterbi decoder uses the Viterbi algorithm for decoding a bitstream that has been encoded using forward error correction based on a convolutional code. There are other algorithms for decoding a convolutionally encoded stream (for example, the Nano algorithm). The Viterbi algorithm is the most resource-consuming, but it does the maximum likelihood decoding.

Here designed the two stage convolutional encoder and viterbi decoder and will implement in FPGA.

For our illustration assume a 4-bit and give as an input to the convolutional encoder rate-1/2 code (two output bits for every input bit). This will yield a 2x4 output matrix, with the extra bits allowing for the correction. This 8 bit output is given as the input to the Viterbi decoder which decodes the convolutional codes into original data.

II. REVIEW OF PREVIOUS ARCHITECTURES

In wireless communication AWGN (Additive White Gaussian Noise) properties of most of the communication media introduce noise in real data during transmission. Channel Coding is a technique to introduce redundant code in real code to remove interference and error during transmission. Coded data in sender side thus increased by volume and error effect becomes less compare with uncoded data. Receiver end receives this data and decodes the data using some techniques. Viterbi decoding is one of the popular techniques to decode data effectively. Viterbi algorithm (VA) is an optimum decoding algorithm for the convolutional code. Convolutional encoding and Viterbi Decoding are widely used for reliable data transmission.

There are different approaches of implementation for Convolutional Encoder and Viterbi Decoder in the literatures. Previously the implementation of Convolutional Encoder and Viterbi Decoder in the DSP Platform. It is a flexible platform but slow in speed. Using μC Platform implementation of Convolutional Encoder and Viterbi Decoder is also slow.

To overcome the performance issue of Convolutional Encoder and Viterbi Decoder, FPGA based implementation has been proposed. These Implementations have Fixed Constraint Length and Code Rate or with Partial Configuration Facility. Complexity of Viterbi decoding algorithm increases in terms of convolutionally encoded trellis length. Increasing the trellis length causes the algorithm to take more time to decode. This will cause transmission speed lower but make the transmission more reliable. Lowering the trellis length will increase the transmission speed. A highly complex Viterbi Decoder someway loses its advantages, when it is adopted to decode sequences transmitted on a low-noise channel. In this case, low minimum distance codes are more suitable for achieving a good performance, and a higher bit rate can be transmitted by lowering the coding rate.

III. CONVOLUTIONAL ENCODER

Convolutional codes are employed to implement FEC convolutional encoder is used to obtain convolution codes .It take a single or multi-bit input and generate a matrix of encoded outputs. In digital modulation communications systems (such as wireless communication systems, etc.) noise and other external factors can alter bit sequences. By adding additional bits make bit error checking more successful and allow for more accurate transfers. By transmitting a greater number of bits than the original signal introduce a certain redundancy that can be used to determine the original signal in the presence of an error.

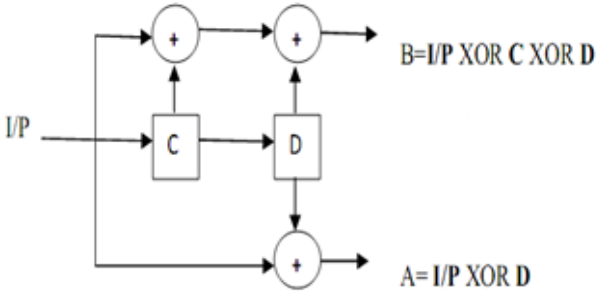
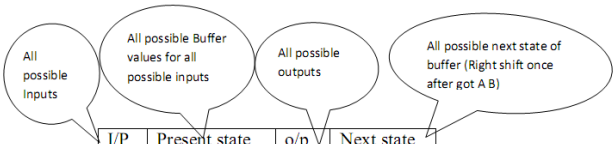


Fig 1: Block Diagram of Convolutional Encoder



I/P	Present state C D	o/p A B	Next state C D
0	00	00	00
0	01	11	00
0	10	01	01
0	11	10	01
1	00	11	10
1	01	00	10
1	10	10	11
1	11	01	11

Fig 2: Truth Table of Convolutional Encoder

A. Trellis Diagram

To draw trellis diagram, refer the above table. Write the all possible present state buffers (C D), then next state (C D). Now refer the above table, for zero input and the buffer values are 00 (C D) the respective next state is 00 (C D). So draw a straight line from 00 of present state to 00 of next state, and above the straight line write the output value (A B) in bracket. As like the same for one input, but the line is dotted line.

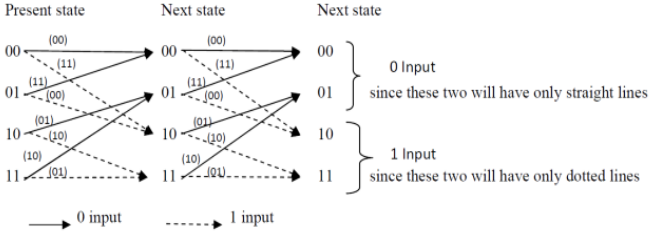


Fig 3: Trellis Diagram

B. State Table

To write the state table, refer the trellis diagram. If the second next state, the 00 is from 00 and 01 of the previous sate. So divide into two states. Write 00 in state 0 and 01 in state 1 and write the respective output (values with in bracket of the straight and dotted line) in value 0 and value 1. As like the same write for 01, 10, 11.Now the required value that is state 0 and 1, Value 0 and 1 are generated.

State 0	State 1	Value 0	Value 1
00	01	00	11
10	11	01	10
00	01	11	00
10	11	10	01

Fig 4: State Table

IV. VITERBI DECODER

Viterbi decoder is most commonly used to resolve convolution codes. This is essential for the purpose of secure transmission of data and its corresponding retrieval during reception. Viterbi decoders also have the property of compressing the number of bits of the data input to half. As a result redundancy in the codes is also reduced. Hence viterbi decoding is more effective and efficient. The Viterbi decoder designed here is an 8:4 decoder. The same logic and concept can also be extended to further number of bits also. Viterbi decoders are based on the basic algorithm which comprises of minimum path and value calculation and retracing the path. This minimum values calculation is determined by EX-OR operation and then comparing .This can be briefed by means of the block diagram.

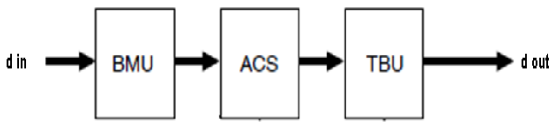


Fig 5: Block Diagram of Viterbi Decoder

BMU: Branch metric unit
ACS: Add, compare and select
TBU: Trace back unit
d in: Input data
d out: Output data

A. Branch Metric Unit

The BMU receives input data from the channel and computes a metric for each state and input combination. The metric is the hamming distance for hard decision encoded data, or l1 norm (sum of absolute values) for soft-decision encoded data. This is used in order to calculate the minimum distance between the 2 given values and the least among the 2 values is chosen. The branch metric unit is the first stage in the implementation of a Viterbi decoder. It is based on the principle of EX ORing the bit values with the values of the 2 respective states. The outputs of the 2 state values are then compared and the resulting minimum value of the two is chosen and taken. Also this incorporates the minimum path and its respective calculation. This path calculation is useful for future.

B. Add, Compare and Select Unit

The ACS unit is the second functional unit of the viterbi decoder. This is based on minimum distance calculations that are obtained from the previous row values. An Add-Compare-Select circuit for use with a trellis decoder can include a first module and a second module. The first module can provide a difference signal specifying an indication of a difference between a second path cost and a first path cost of a trellis. The second path cost can be a sum of a second state cost and a second branch metric and the first path cost can be a sum of a first state cost and a first branch metric. The ACS computation compares the sampled symbol value with the values that would be expected for each possible transition on a noiseless channel. The metric is the distance (hamming distance for hard-decision, and euclidean distance for soft-decision decoders) from the actual symbol to an expected symbol; the smallest metric indicates the closest match. The ACS adds the current metric to the accumulated metric for each path and determines the least metric for each state of the trellis. The ACS retrieves the accumulated metric from the register files, and then adds the current metric. The traceback unit traces back the trellis after a block of data (determined by the trace back length) has been processed by the ACS. The ACS unit after computation for a row moves forward to all the other rows and the implementation continues. For a [n: k] decoder, the ACS operates for k-1 times. Thus in the [8:4] Viterbi decoder that use, the ACS is implemented 3 times.

C. Trace Back Unit

Back-trace unit restores an (almost) maximum-likelihood path from the decisions made by PMU. This is the final stage of the Viterbi decoder where the input that was transmitted by using the convolution encoder is once again retrieved and the 4 bit message is obtained. Trace back unit traces from the final row upto the first and hence a bit reversal is effected in the end so the four bits are obtained in the correct order. First, the TBU establishes an optimal path by starting from the node of minimum metric and traces back the path in the trellis all the way to the beginning of the trellis tree. Then, the original

data corresponding to the encoded data is determined. Reverse transpose operations that depend on the last active add-compare-select unit a cascade block of the state metric update process. One important criterion for construction of the TBU is the position of the various minimum distance values. As a result the value corresponding to the minimum path is chosen and depending the output is obtained.

D. Viterbi Algorithm

The algorithm for Viterbi decoder can be explained by using the example below. Let us consider the input 11010001.

Minimum value and minimum path calculation

The input is taken in the form of four sets each, comprising of 2 bits starting from the MSB. The first two bits from the MSB are taken and are XOR-ed with the values predetermined for the encoder as STATE 0 and the values are obtained. Similarly they are also EXOR-ed with the values of STATE 1 and the values are also noted down. Thus obtain 2 sets of 4 values each, for state 0 and state 1. The corresponding values are compared and the minimum of each of the 2 values is noted down for all the 4 values. Also corresponding position values are marked as 00,01,10,11 respectively. For the minimum path calculation, the value corresponding to that particular minimum path is taken and its state value (state 0 or state 1) is assigned to the minimum path in decimal notation. This is the function of the BMU.

Value 0	Value 1	Min Distance	Path
11 Xor 00 = 11= 3	11 Xor 11 = 00 = 0	0 (00)	1
11 Xor 01 = 10 = 2	11 Xor 10 = 01 = 1	1 (01)	3
11 Xor 11 = 00 = 0	11 Xor 00 = 11= 3	0 (10)	0
11 Xor 10 = 01 = 1	11 Xor 01 = 10 = 2	1 (11)	2
01 Xor 00 = 01 = 1+0=1	01 Xor 11 = 10 = 2+1=3	1 (00)	0
01 Xor 01 = 00 = 0+0=0	01 Xor 10 = 11 = 3+1=4	0 (01)	2
01 Xor 11 = 10 = 2+0=2	01 Xor 00 = 01 = 1+1=2	2 (10)	0 or 1
01 Xor 10 = 11 = 3+0=3	01 Xor 01 = 00 = 0+1=1	1 (11)	3
00 Xor 00 = 00 = 0+1=1	00 Xor 11 = 11 = 3+0=3	1 (00)	0
00 Xor 01 = 01 = 1+2=3	00 Xor 10 = 10= 2+1=3	3 (01)	2 or 3
00 Xor 11 = 11 = 3+1=4	00 Xor 00 = 00= 0+0=0	0 (10)	1
00 Xor 10 = 10 = 2+2=4	00 Xor 01 = 01= 1+1=2	2 (11)	3
01 Xor 00 = 01 = 1+1=2	01 Xor 11 = 10 = 2+3=5	2 (00)	0
01 Xor 01 = 00 = 0+0=0	01 Xor 10 = 11 = 3+2=5	0 (01)	2
01 Xor 11 = 10 = 2+1=3	01 Xor 00 = 01 = 1+3=4	3 (10)	0
01 Xor 10 = 11 = 3+0=3	01 Xor 01 = 00 = 0+2=2	2 (11)	3

Once the 1st row is resolved, the second set of two bits starting from the MSB (01 in our case) is EXOR-ed as explained for the first set. Now the corresponding states of the respective values (value 0 or value 1) are obtained and their corresponding STATE VALUES are noted down (in binary). These state values denote the minimum distance positions, obtained in the previous row resolution. The corresponding value of the minimum distance is chosen and is added to the EXOR-ed outputs of the present row (both in binary) for both value 0 and value 1. They are then compared and minimum distance is once again obtained and corresponding positions are assigned. This is the function of the ACS unit. The path calculation is the same as the first row.

The above procedure is repeated for rows 3 and 4 also and their corresponding values are obtained.

Path trace back procedure

The final step is the trace back procedure, wherein all the values are consolidated to obtain the final output. If the position of minimum value is 00 or 01, a 0 is obtained in the output. For any other values, a 1 is obtained at the output. In this calculation, the minimum value of the last row is taken and based on its position a 1 or 0 is assigned as one of the output bits. The corresponding path is noted and converted to binary. The minimum value corresponding to that particular path is noted and once again a position based output bit assignment is made as explained previously, for the row 3. This is repeated for all the remaining rows. Thus, 4 output bits are obtained. These bits are then concatenated and a process of bit reversal is made. Thus a 4 bit output is derived from an 8 bits input that was given to the viterbi decoder.

The output for the 8 to 4 viterbi decoder is shown below. The encoded bit is given as input and the 4 bit original message bit is decoded and got as the output.

V. SIMULATIONS

The simulations of the designs are performed using MODELSIM and synthesis report for using Xilinx. Modelsim waveform & Synthesis Report of convolutional encoder and viterbi decoder:

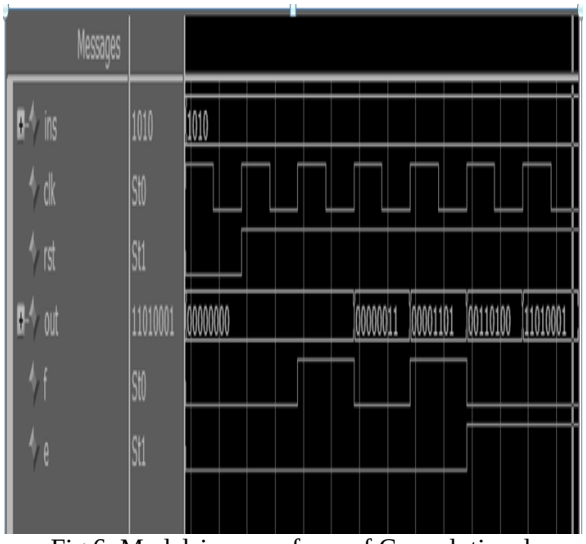


Fig 6: Modelsim waveform of Convolutional Encoder

```
=========================================================
*                Final Report                           *
=========================================================
Final Results
RTL Top Level Output File Name     : top_test.ngr
Top Level Output File Name         : top_test
Output Format                      : NGC
Optimization Goal                  : Speed
Keep Hierarchy                     : NO

Design Statistics
# IOs                              : 14

Macro Statistics :
# Registers                        : 8
#      1-bit register              : 2
#      2-bit register              : 5
#      3-bit register              : 1
# Tristates                        : 1
#      2-bit tristate buffer       : 1

Cell Usage :
# BELS                             : 24
#      GND                         : 1
#      LUT1                        : 1
#      LUT1_L                      : 1
#      LUT2                        : 6
#      LUT2_L                      : 1
#      LUT3_D                      : 1
#      LUT3_L                      : 5
#      LUT4                        : 6
#      LUT4_L                      : 1
#      VCC                         : 1
# FlipFlops/Latches                : 29
#      FDC                         : 20
#      FDCE                        : 4
#      FDE                         : 1
#      FDP                         : 1
#      LD                          : 3
# Tri-States                       : 2
#      BUFT                        : 2
# Clock Buffers                    : 1
#      BUFGP                       : 1
# IO Buffers                       : 13
#      IBUF                        : 5
```

Fig 7: Synthesis Report of Convolutional Encoder

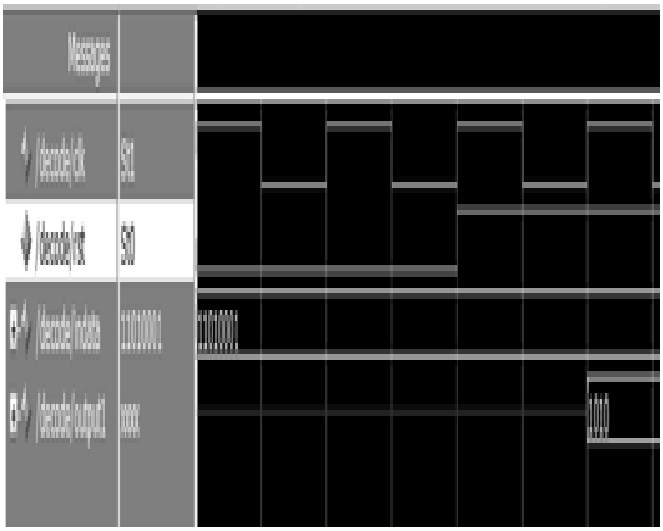


Fig 8: Modelsim waveform of Viterbi Decoder

```
=========================================================
*                Final Report                           *
=========================================================
Final Results
RTL Top Level Output File Name     : decode.ngr
Top Level Output File Name         : decode
Output Format                      : NGC
Optimization Goal                  : Speed
Keep Hierarchy                     : NO

Design Statistics
# IOs                              : 14

Macro Statistics :
# Registers                        : 1
#      1-bit register              : 1
# Multiplexers                     : 22
#      2-to-1 multiplexer          : 22
# Comparators                      : 12
#      3-bit comparator less       : 12
# Xors                             : 24
#      1-bit xor3                  : 24

Cell Usage :
# BELS                             : 184
#      LUT1                        : 9
#      LUT2                        : 17
#      LUT3                        : 37
#      LUT4                        : 120
#      VCC                         : 1
# FlipFlops/Latches                : 17
#      FDC                         : 1
#      LD                          : 16
# Clock Buffers                    : 1
#      BUFGP                       : 1
# IO Buffers                       : 13
#      IBUF                        : 9
#      OBUF                        : 4
```

Fig 9: Synthesis Report of Viterbi Decoder

VI. CONCLUSION

In this paper, a Viterbi algorithm based on the strongly connected trellis decoding of binary convolutional codes has been implemented in FPGA. The use of error-correcting codes has proven to be an effective way to overcome data corruption in digital communication channels. The adaptive Viterbi decoders are modeled using Verilog, and post synthesized by Xilinx FPGA logic.

Thus implement a higher performance Viterbi decoder with such as pipelining or interleaving. So in the future, with Pipeline or interleave the ACS and the trace-back and output decode block, make it better.

REFERENCES

[1] Shannon, C. "A mathematical theory of Communication", Bell Sys.Tech .J, vol. 27, pp. 379-423 and 23-656, 1948.

[2] Hamming, R. "Error detecting and correcting codes", Bell Sys.Tech .J, vol. 29, pp.147-160, 1960.

[3] Golay, M. "Notes on digital coding", Proc. IEEE, Vol. 37, p. 657 , 1949 .

[4] Azim, C., Monir, M. "Implementation of Viterbi Decoder for WCDMA System", Proceedings of IEEE International Conference (NMIC 2005), pp. 1-3, 2005.

[5] Wong, Y., Jian, W., HuiChong, O., Kyun, C., Noordi, N. "Implementation of Convolutional Encoder and Viterbi Decoder using VHDL", Proceedings of IEEE International conference on Research and Development Malaysia, November 2009.

[6] Bissi, L., Placidi. P., Baruffa, G., Scorzoni, A. "A Multi- Standard Reconfigurable Viterbi Decoder using Embedded FPGA blocks", Proceedings of the 9th EUROMICRO Conference on Digital System Design(DSD'06),pp. 146-153 , 2006.

[7] Shaker, S., Elramly. S, Shehata. K."FPGA Implementation of a reconfigurable Viterbi Decoder for WiMax Receiver", IEEE International conference on Microelectronics, pp. 246-267, 2009.

[8] F. Chan and D. Haccoun, "Adaptive viterbi decoding of convolutional codes over memoryless channels," IEEE Trans. Commun., vol. 45, no.11, pp. 1389–1400, Nov. 1997.

[9] Liakot Ali, Mohammad Bozlul Karim, "Simulation and Design of Parameterized Convolutional Encoder and Viterbi Decoder for Wireless Communication Md. S.M Tofayel Ahmad Department of Information and Communication Technology Institute of Information and Communication Technology Bangladesh University of Engineering and Technology, Bangladesh.