Design and Comparison of Viterbi Decoder on Spartan-3A (XC3S400A- 4FTG256C) and Spartan-3E (XC3S500E- 4FT256) Using Verilog

[1]Jigar B Patel, [2]Prof.Nabila Shaikh
[1]L.J. Institute of Engineering and Technology, GTU, Gujarat, India
[2]Gandhinagar Institute Of Technology, GTU, Gujarat, India
[1]Jigar1558@gmail.com

***Abstract*- It is well known that data transmissions over wireless channels are affected by attenuation, distortion, interference and noise, which affect the receiver's ability to receive correct information. Convolutional encoding with Viterbi decoding is a powerful method for forward error detection and correction. It has been widely deployed in many wireless communication systems to improve the limited capacity of the communication channels. In this paper, we present a comparison of Two different method for Implementation which is Spartan-3A (XC3S400A- 4FTG256C) Field-Programmable Gate Array and Spartan-3E (XC3S500E- 4FT256) of Viterbi Decoder with a constraint length of 3 and a code rate of 1/2.**

***Keywords*- Convolutional encoder, FPGA, Traceback method, Spartan-3A (XC3S400A- 4FTG256C) Board, Spartan-3E (XC3S500E- 4FT256) , Viterbi decoder**

I. INTRODUCTION

With the growing use of digital communication, there has been an increased interest in high-speed Viterbi decoder design within a single chip. Advanced field programmable gate array (FPGA) technologies and well developed electronic design automatic (EDA) tools have made it possible to realize a Viterbi decoder with the throughput at the order of Giga-bit per second, without using off-chip processor(s) or memory.

Motivation for low power has been derived from needs to increase the speed, to extend the battery life and to reduce the cost along with this the main objectives of this paper is to design an efficient Decoder by providing Efficient Decoding by providing the most perfect predictable output that was most likely to be transmitted, Fixed Decoding Time, Increasing the Efficiency of the System, Eliminating the effect of noise in a wide variety of system including Mobile, Satellite communication etc Simple to Implement.

The A.J. Viterbi developed an asymptotically optimal decoding algorithm for convolutional codes. The Viterbi Algorithm, the elegant 41-year-old logical tool for rapidly eliminating dead end possibilities in data transmission, has a new application to go alongside its ubiquitous daily use in Cell Phone Communications, Bioinformatics, Speech Recognition and many other areas of Information Technology. Viterbi Decoding has the advantage of fixe decoding time. It is well suited to Hardware implementation.

II. VITERBI DECODER

A structure and short overview of the basic Viterbi decoding system is illustrated in Fig. 1. This figure shows three basic elements of the Viterbi decoding communication system: convolutional encoder, communication channel and Viterbi decoder.

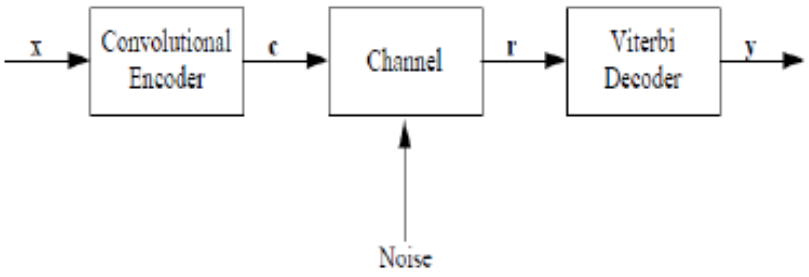

Figure 1 Viterbi Decoder Communication system

Convolutional Encoder

Convolutional code is a type of error-correcting code in which each (n_m) m-bit information symbol (each mbit string) to be encoded is transformed into an n-bit symbol, where m/n is the code rate (n_m) and the transformation is a function of the last k information symbols, where K is the constraint length of the code.

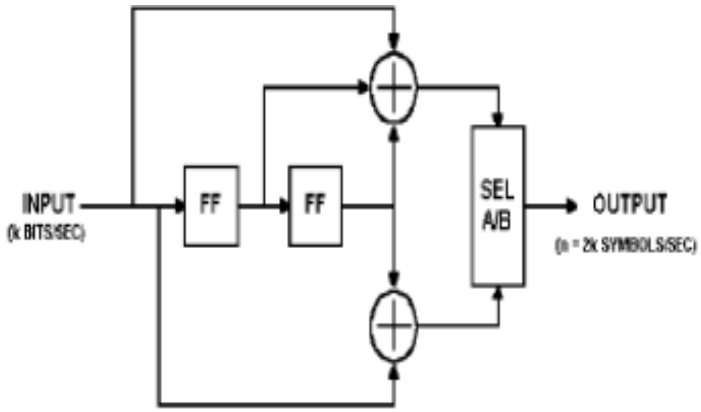

Figure 2 the rate ½ Convolutional Encoder

To convolutional encoded data, start with k memory registers, each holding 1 input bit. Unless otherwise specified, all memory registers start with a value of 0.The encoder has n modulo-2 adders, and n generator polynomials—one for each adder (see figure1).An input bit m1 is fed into the leftmost register. Using the generator polynomials and the existing values in the remaining registers, the encoder outputs n bits.

Viterbi Algorithm

A. J. Viterbi proposed an algorithm as an 'asymptotically optimum' approach to the decoding of convolutional codes in memory-less noise. The Viterbi algorithm (VA) is knows as a maximum likelihood (ML)-decoding algorithm for convolutional codes. Maximum likelihood decoding means finding the code branch in the code trellis that was most likely to be transmitted.

Therefore, maximum likelihood decoding is based on calculating the hamming distances for each branch forming encode word. The most likely path through the trellis will maximize this metric. [7] Viterbi algorithm performs ML decoding by reducing its complexity. It eliminates least likely trellis path at each transmission stage and reduce decoding complexity with early rejection of unlike pathes.Viterbi algorithm gets its efficiency via concentrating on survival paths of the trellis. The Viterbi algorithm is an optimum algorithm for estimating the state sequence of a finite state process, given a set of noisy observations. [2] The implementation of the VA consists of three parts: branch metric computation, path metric updating, and survivor sequence generation. The path metric computation unit computes a number of recursive equations. In a Viterbi decoder (VD) for an N-state convolutional code, N recursive equations are computed at each time step (N = 2k-1, k= constraint length). Existing high-speed architectures use one processor per recursion equation. The main drawback of these Viterbi Decoders is that they are very expensive in terms of chip area. In current implementations, at least a single chip is dedicated to the hardware realization of the Viterbi decoding algorithm the novel scheduling scheme allows cutting back chip area dramatically with almost no loss in computation speed.

Viterbi Decoder

The basic units of Viterbi decoder are branch metric unit, add compare and select unit and survivor memory management unit.

1. Branch Metric Unit
   The first unit is called branch metric unit. Here the received data symbols are compared to the ideal outputs of the encoder from the transmitter and branch metric is calculated. Hamming distance or the Euclidean distance is used for branch metric computation.
2. Path Metric Unit
   The second unit, called path metric computation unit, calculates the path metrics of a stage by adding the branch metrics, associated with a received symbol, to the path metrics from the previous stage of the trellis.

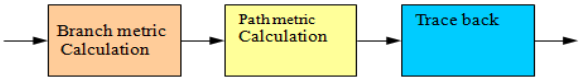

Figure: Block Diagram of Viterbi decoder

3. Survivors Memory Management Unit
   The final unit is the trace-back process or register exchange method, where the survivor path and the output data are identified. The trace-back (TB) and the register exchange (RE) methods are the two major techniques used for the path history management in the chip designs of Viterbi decoders. The TB method takes up less area but requires more time as compared to RE method because it needs to search or trace the survivor path back sequentially. Also, extra hardware is required to reverse the decoded bits. The major disadvantage of the RE approach is that its routing cost is very high especially in the case of long constraint lengths and it requires much more resources.
4. Traceback Method and Register Exchange method
   In the TB method, the storage can be implemented as RAM and is called the path memory. Comparisons in the ACS unit and not the actual survivors are stored. After at least L branches have been processed, the trellis connections are recalled in the reverse order and the path is traced back through the trellis diagram The TB method extracts the decoded bits, beginning

from the state with the minimum PM. Beginning at this state and tracing backward in time by following the survivor path, which originally contributed to the current PM, a unique path is identified. While tracing back through the trellis, the decoded output sequence, corresponding to the traced branches, is generated in the reverse order. Trace back architecture has a limited memory bandwidth in nature, and thus limits the decoding speed The register exchange (RE) method is the simplest conceptually and a commonly used technique. Because of the large power consumption and large area required in VLSI implementations of the RE method, the trace back method (TB) method is the preferred method in the design of large constraint length, high performance Viterbi decoders[1]. In the register exchange, a register assigned to each state contains information bits for the survivor path from the initial state to the current state. In fact, the register keeps the partially decoded output sequence along the path. The register of state S1 at t=3 contains '101'. This is the decoded output sequence along the hold path from the initial state.

III. CODING

VHDL is the VHSIC Hardware Description Language. VHSIC is an abbreviation for Very High Speed Integrated Circuit. It can describe the behaviour and structure of electronic systems, but is particularly suited as a language to describe the structure and behaviour of digital electronic hardware designs, such as ASICs and FPGAs as well as conventional digital circuits.Using Hardware Description Languages (HDLs) to design high-density FPGA devices has the advantages of Top-Down Approach for Large Projects, Functional Simulation Early in the Design Flow, Synthesis of HDL Code to Gates The Behavioural Verilog module describes features of the language that describe the behaviour of components in response to signals. Behavioural descriptions of hardware utilize software engineering practices and constructs to achieve a functional model. Timing information is not necessary in a behavioural description, although such information may be included easily. The Verilog process construct is described first. Processes run code sequentially.The statements allowed in a process, referred to as 'sequential' statements, are listed in the module. The Behavioural Verilog module ends with a comprehensive example using the quick sort routine. Although a detailed understanding of the algorithm implemented by this routine are not important for a full understanding of the Verilog constructs presented in this module, the example serves as a vehicle for highlighting many of the Verilog features presented in this module. The model also illustrates the similarity between processoriented Verilog descriptions and other general-purpose high-level programming languages.

Vietrbi decoder algorithm Design flow

The algorithm can be broken down into the following three steps.

1. Weigh the trellis; that is, calculate the branch metrics.
2. Recursively computes the shortest paths to time n, in terms of the shortest paths to time n-1. In this step, decisions are used to recursively update the survivor path of the signal. This is known as add-compareselect (ACS) recursion.

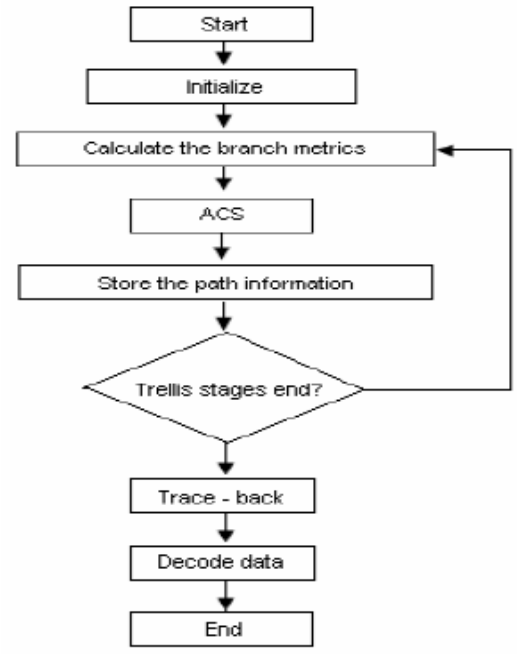

Figure: Viterbi decoder algorithm Design flow

3. Recursively finds the shortest path leading to each trellis state using the decisions from Step 2. The shortest path is called the survivor path for that state and the process is referred to as survivor path decode. Finally, if all survivor paths are traced back in time, they merge into a unique path, which is the most likely signal path.

IV. IMPLEMENTATION AND RESULTS

Device implementation is done to put a verified code on FPGA. The various steps in design implementation are:

1. Translate
2. Map
3. Place and route
4. Configure

The full design flow is an iterative process of entering, implementing, and verifying your design until it is correct and complete. The Xilinx Development System allows quick design iterations through the design flow cycle. Xilinx devices permit unlimited reprogramming.

A. Implementation Results

System level Verification

In system level verification the 8 bit binary input will be feed through DIP switches to convolutional encoder which produces a 16 bit encoded output

DIP switch output: 10101101

Convolution encoder o/p: 0101110010100010

convolutional encoder output will be feed to FPGA

through 2:1 multiplexer the FPGA performs the viterbi decoding of this data by utilizing maximum likelihood method and recovers the output that is most likely as it was been transmitted.

FPGA output: 10101101

Experimental Analysis of Viterbi Decoder Implementation using Traceback method on The developed Viterbi Decoder using Traceback method is implemented using Verilog and Synthesized and deployed on SPARTAN 3A XC3S400A target FPGA platform. The implemented module is tested and verified by simulation and the area utilization of FPGA is evaluated through the Synthesis Report

B. Simulation result

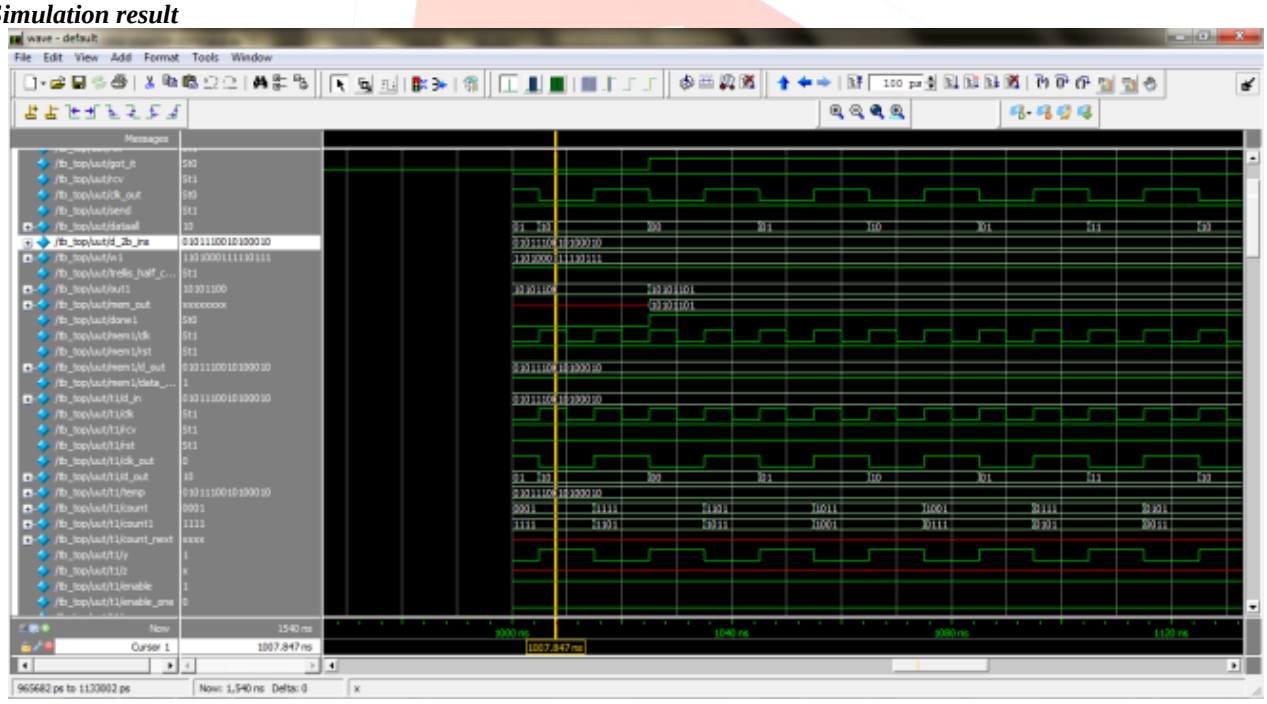

Figure 3 Simulation waveforms

RTL Schematic

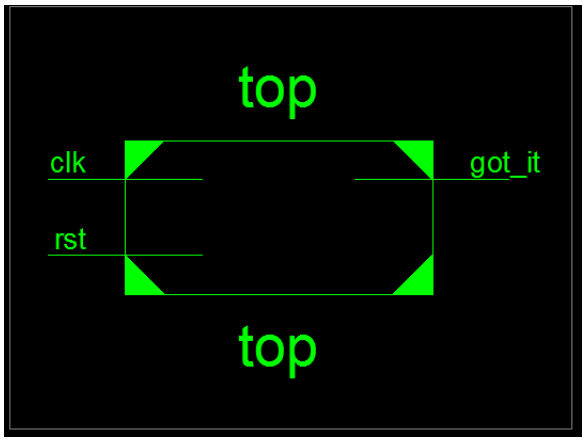

Figure 4 RTL Schematic of Viterbi Decoder using Traceback Method

Advanced HDL Synthesis Report of spartan 3A

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Flip Flops	35	7,168	1%
Number of 4 input LUTs	47	7,168	1%
Number of occupied Slices	36	3,584	1%
Number of Slices containing only related logic	36	36	100%
Number of Slices containing unrelated logic	0	36	0%
Total Number of 4 input LUTs	47	7,168	1%
Number of bonded IOBs	3	195	1%
IOB Flip Flops	1		
Number of BUFGMUXs	1	24	4%
Average Fanout of Non-Clock Nets	3.84		

Advanced HDL Synthesis Report of spartan 3E

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Flip Flops	35	9,312	1%
Number of 4 input LUTs	46	9,312	1%
Number of occupied Slices	36	4,656	1%
Number of Slices containing only related logic	36	36	100%
Number of Slices containing unrelated logic	0	36	0%
Total Number of 4 input LUTs	46	9,312	1%
Number of bonded IOBs	3	190	1%
IOB Flip Flops	1		
Number of BUFGMUXs	1	24	4%
Average Fanout of Non-Clock Nets	3.78		

C. ***Comparison of Experimental and Analytical Methods***

Sr. No.	Parameter	Using spartan 3A / Available	Using spartan 3E/Available
1	Number of Slice Flip Flops	35/7168	35/9312
2	Number of 4 input LUTs	47/7168	46/9312
3	Number of bonded IOBs	3/195	3/190
4	Number of GCLKs	001/24	001/24
5	Minimum Perioid	4.075	4.668
6	Frequency	243.399 MHz	213.311MHz

Hence because of the large power consumption and large area required in VLSI implementations using FPGA is used.

V. CONCLUSION

In this paper we presented an implementation of the Viterbi Decoder with constraint length of 3 and code rate of ½, The proposed solution has proven to be particularly efficient in terms of the required FPGA implementation resources so as Chip Silicon Area, Decoding Time and Power Consumption. We have developed Viterbi Decoder on Spartan 3A FPGA and also on Spartan 3E FPGA by utilizing traceback method and Synthesis result shows that Spartan 3E field programmable gate array method is more efficient compare to another in term of Chip Area Utilization and it have a less frequency so as will be less Power Consumption in comparision with Spartan 3A Method. We have also tested the functionality of the Viterbi Decoder Code implemented on FPGA by designing a Test Bench for performing Error Detection and Correction.

VI. REFERENCES

[1] Vasily P. Pribylov, Alexander I. Plyasunov (2005). "A Convolutional Code Decoder Design Using Viterbi Algorithm with Register Exchange History Unit". SIBCON. IEEE.

[2] John G. Proakis (2001). "Digital Communication". McGraw Hill, Singapore. Pp 502- 507, 471-475

[3] S. K. Hasnain, Azam Beg and S. M. Ghazanfar Monir (2004). "Performance Analysis of Viterbi Decoder Using a DSP Technique". INMIC. IEEE. pp 201-206.

[4] Irwin M. Jacobs (1974). "Practical Applications of Coding". IEEE. pp 305-310.

[5] Xun Liu Marios C. Papaefthymiou. "Design of a High-Throughput Low-PowerIS95 Viterbi Decoder". Advanced Computer Architecture Laboratory, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, Michigan.

[6] Tommy Oberg (2001). "Modulation Detection and Coding". Wiley and Sons. Pp 81- 86.

[7] YOU Yu-xin, WANG Jin-xiang, LAI Feng-chang and YE Yi-zheng (2002)."VLSI Design and Implementation of High Speed Viterbi Decoder". IEEE .pp 64-66.

[8] Stefan Bitterlich and Heinrich Meyr (1993). "Efficient Scalable Architectures for Viterbi Decoders". Aachen University of Technology, Templergraben , Germany. pp 89-100.

[9] Sang-Cheon Kim, Je-Hyuk Ryu, and Jun-Dong Cho. "Low Power High-Rate Viterbi Decoder Employing the SST (Scarce State Transition) Scheme and Radix- 4 Trellis" Department of Electrical and Computer Engineering, Sungkyunkwan University.

[10] KIM MIM WOO (2005). "High-speed Viterbi Decoder Architecture".

[11] I. Bogdan, M. Munteanu, P. A. Ivey, N. L. Seed, N. Powell. "Power Reduction Techniques for a Viterbi Decoder Implementation". Electronic Systems Group, University of Sheffield, Mappin Street, Sheffield S1 3EA, UK

[12] Samirkumar Ranpara and Dong Sam Ha. "Low-Power Viterbi Decoder Design for Wireless Communications Applications".