FPGA Based Efficient Implementation of Viterbi Decoder

Anubhuti Khare, Manish Saxena, Jagdish Patel,

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 Spartan XC3S400A Field-Programmable Gate Array efficient implementation of Viterbi Decoder with a constraint length of 3 and a code rate of 1/3. The Viterbi Decoder is compatible with many common standards, such as DVB, 3GPP2, 3GPP LTE, IEEE 802.16, Hiperlan, and Intelsat IESS-308/309.***

***Keywords*—Convolutional encoder, FPGA, Register Exchange, Spartan XC3S400A Board, 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 fixed decoding time. It is well suited to Hardware implementation.

Manuscript received October 06, 2011.

Dr. Anubhuti Khare, Reader,Department of Electronics and Communication, University Institute of Technology, Rajeev Gandhi Technical University, Bhopal, (MP), India (Email:-anubhutikhare@gmail.com, Mobile:+919425606502).

Manish Saxena, Head of Electronics and Communication Department, Bansal Institute Of Science and Technology, Bhopal, (MP), India (Email:-manish.saxena2008@gmail.com, Mobile: +919826526247)

Jagdish Patel, M.Tech (Digital Comm), Bansal Institute of Science and Technology, Bhopal, (MP), India (Email:-Jagdishpatel07@gmail.com, Mobile:+ 918793495989, 09404816550)

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.

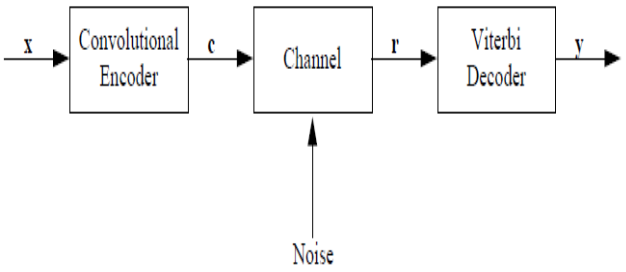

Figure: Viterbi Decoder Communiction system

A. 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.

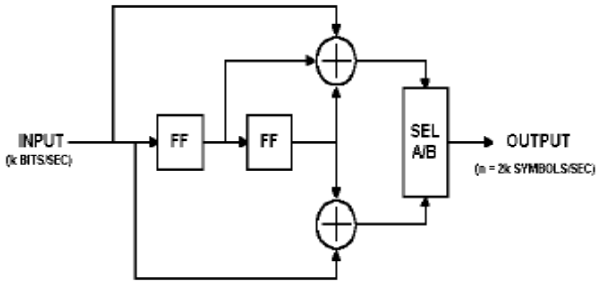

Figure: The rate ½ Convolutional Encoder

To convolutionally 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

B. 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.

C. 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

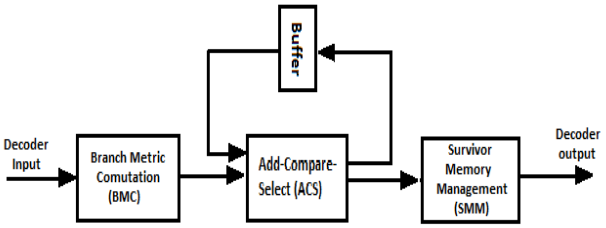

Figure: Block Diagram of Viterbi decoder

branch metrics, associated with a received symbol, to the path metrics from the previous stage of the trellis

3) Survivour 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. PROGRAMMABLE DEVICES

Programmable devices are those devices which can be programmed by the user. Various programmable devices are PLDs, CPLDs, ASICs and FPGAs.

A. Field Programmable Gate Arrays

'Field Programmable' means that the FPGA's function is defined by a user's program rather than by the manufacturer of the device. A Field Programmable Gate Array (FPGA) is a semiconductor device containing programmable logic components and programmable interconnects. The programmable logic components can be programmed to duplicate the functionality of basic logic gates such as AND, OR, XOR, NOT or more complex combinational functions such as decoders or simple math functions. In most FPGAs, these programmable logic components (or logic blocks, in FPGA parlance) also include memory elements, which may be simple flip-flops or more complete blocks of memories. Each process is assigned to a different block of the FPGA and operates independently.

FPGAs originally began as competitors to CPLDs and competed in a similar space, that of glue logic for PCBs. As their size, capabilities and speed increase, they began to take over larger and larger functions to the state where they are now market as competitors for full systems on chips. They now find applications in any area or algorithm that can make use of the massive parallelism offered by their architecture. [2]

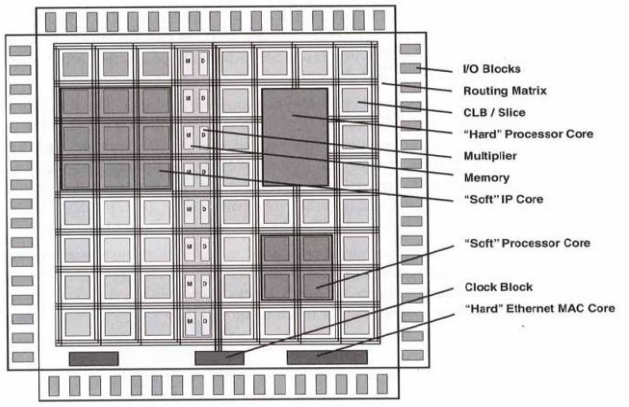

Figure: FPGA Internal Architecture

B. SPARTAN XC3S400A FPGA

The Spartan®-3A family of Field-Programmable Gate Arrays (FPGAs) solves the design challenges in most high-volume, cost-sensitive, I/O-intensive electronic applications.Because of their exceptionally low cost, Spartan-3A FPGAs are ideally suited to a wide range of consumer electronics applications, including broadband access, home networking, display/projection, and digital television equipment. A Xilinx Spartan-3A (XC3S400A-4FTG256C) 400 K gate FPGA and a Cypress Cy8C24894 PSoC Mixed-Signal Array are the primary components of the Avnet Spartan-3A evaluation board. In addition to on-board processing functions, the PSoC device provides offboard communication via a USB 2.0 full-speed interface.

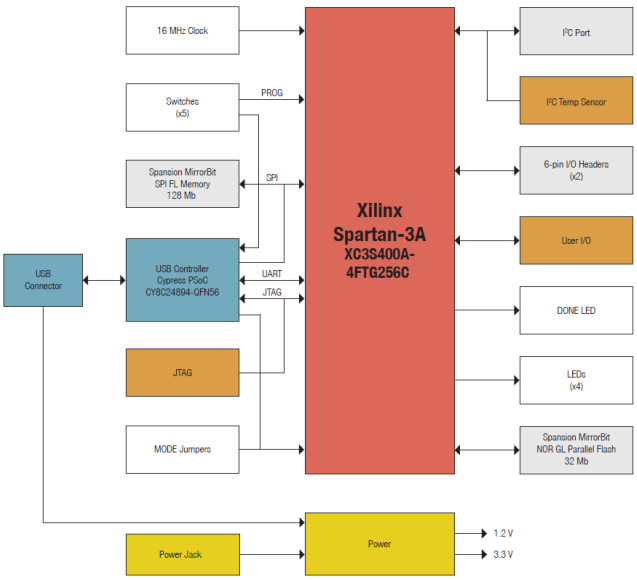

Figure: Spartan-3A Evaluation Board Block Diagram

IV. VITERBI DECODER DESIGN AND IMPLEMENTATION

A. System block diagram

The figure 6 shows the hardware design of a viterbi decoder. The convolutional encoder is realized by a hardware using XOR Gate and Flip Flop. The 8 bit input is given through a DIP switches the shift register is used to serialized and provide this input to convolutional encoder the output of encoder is given to a SPARTAN FPGA through a 2:1 multiplexer for performing viterbi decoding.

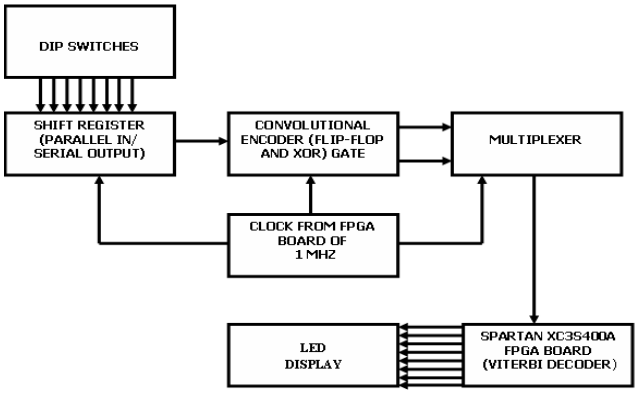

Figure: System Block Diagram

Table: Encoder parameter

parameter	Value(bits)
Input	1
Output	2
Code rate	½
Constraint length	3

As said earlier there are two methods for performing viterbi decoding so as register exchange and traceback method. In this project both method are implemented on to FPGA board and results of two are compared.

B. 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 VHDL 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 VHDL 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 VHDL 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 VHDL constructs presented in this module, the example serves as a vehicle for highlighting many of the VHDL features presented in this module. The model also illustrates the similarity between process-oriented VHDL descriptions and other general-purpose high-level programming languages.

C. 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-compare-select (ACS) recursion.

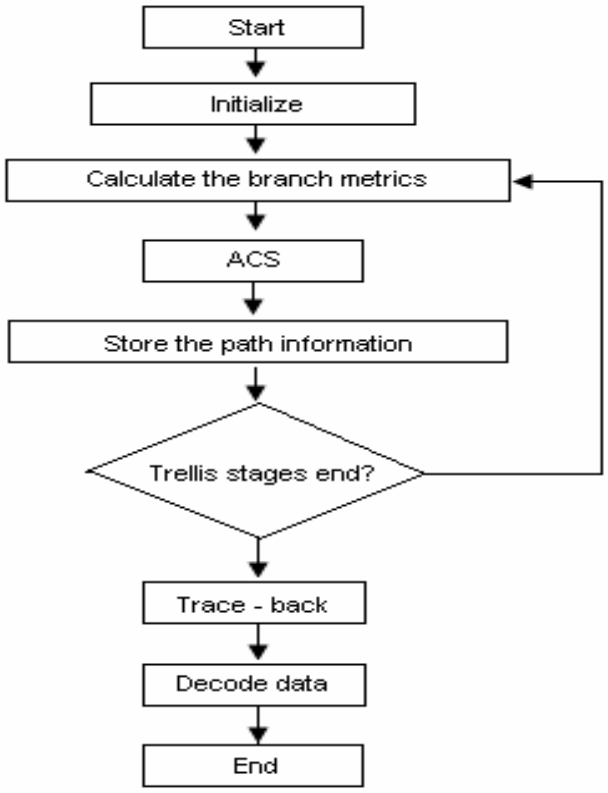

Figure: Viterbi decoder algorithm Design flow

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

D. Design entry using xilinx ISE 10.1 design Suite

In the design entry process, the behavior of circuit is written in hardware description language like VHDL. Simulation and synthesis are the two main kinds of tools which operate on the VHDL language. VHDL does not constrain the user to one style of description. VHDL allows designs to be described using any methodology - top down or bottom up. VHDL can be used to describe hardware at the gate level or in a more abstract way. Xilinx ISE 10.1 design Suite these software manuals support the Xilinx® Integrated Software Environment (ISE™) software complete design entry is to be done by using the same software. Xilinx maintains software libraries with hundreds of functional design elements (unimacros and primitives) for different device architectures

1) Synthesis

First, an intermediate representation of the hardware design is produced. This step is called synthesis and the

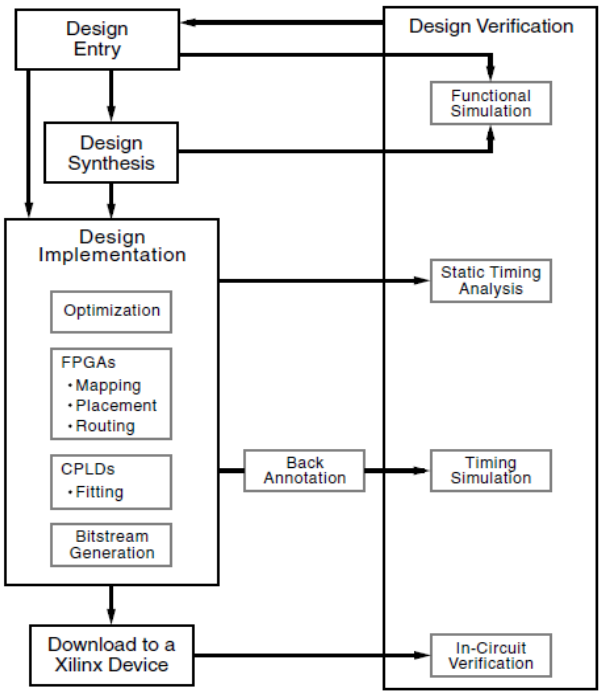

Figure: Xilinx Design Flow

result is a representation called a netlist. In this step, any semantic and syntax errors are checked. The synthesis report is created which gives the details of errors and warning if any. The netlist is device independent, so its contents do not depend on the particulars of the FPGA or CPLD; it is usually stored in a standard format called the Electronic Design Interchange Format (EDIF).

2) Simulation

Simulator is a software program to verify functionality of a circuit. The functionality of code is checked. The inputs are applied and corresponding outputs are checked. If the expected outputs are obtained then the circuit design is correct. Simulation gives the output waveforms in form of zeros and ones. Although problems with the size or timing of the hardware may still crop up later, the designer can at least be sure that his logic is functionally correct before going on to the next stage of development.

3) Implementation

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.

V. IMPLEMENTATION RESULTS

A. 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: 0 1 1 0 1 0 0 0

Convolution encoder o/p: 00 11 00 01 01 11 10 01

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: 0 1 1 0 1 0 0 0

B. Experimental Analysis of Viterbi Decoder Implementation using Traceback method:

The developed Viterbi Decoder using Traceback method is implemented using VHDL 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

1) simulation result

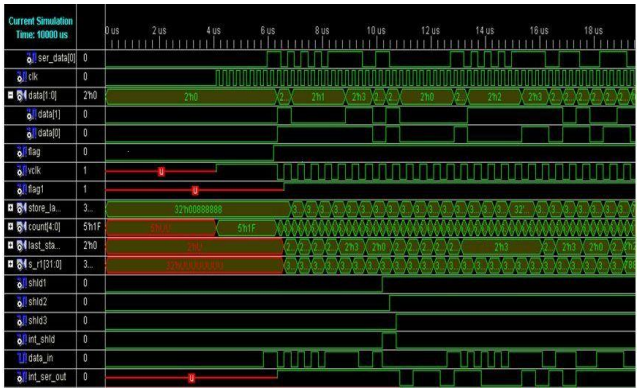

Figure: Simulation waveforms

2) RTL Schematic

Figure 9: RTL Schematic of Viterbi Decoder using Traceback Method

3) Advanced HDL Synthesis Report

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Total Number Slice Registers	98	7,168	1%
Number used as Flip Flops	97		
Number used as Latches	1		
Number of 4 input LUTs	182	7,168	2%
Logic Distribution			
Number of occupied Slices	133	3,584	3%
Number of Slices containing only related logic	133	133	100%
Number of Slices containing unrelated logic	0	133	0%
Total Number of 4 input LUTs	185	7,168	2%
Number used as logic	182		
Number used as a route-thru	3		
Number of bonded IOBs	14	195	7%
Number of BUFGMUXs	2	24	8%

4) Testing of Viterbi Decoder by Traceback Method under Noisy Conditions

The functional verification of Viterbi Decoder code on FPGA done by designing a Test Bench. The Simulation result shown in waveforms below has two resultant output waveform for input stimulus i.e. Convolutional Encoded data without error and other for Convolutional Encoded data with error. This Simulation result is based on the Test Bench designed to test the ability of Viterbi Decoder code to Detect and Correct the Error. Figure shows that output of viterbi decoder is same for both.

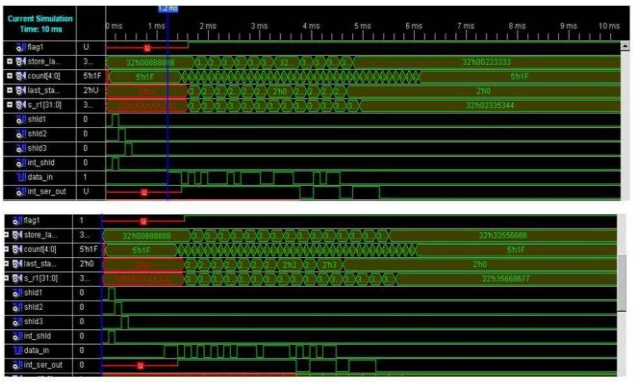

Figure: Test Bench Simulation waveforms

C. Analytical Verification of Viterbi Decoder Implementation using Register Exchange method

1) RTL Schematic€

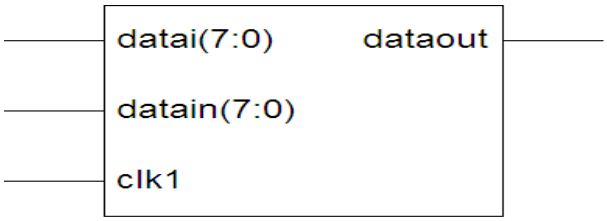

Figure: RTL Schematic

2) Advanced HDL Synthesis Report

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Flip Flops	1,910	7,168	26%
Number of 4 input LUTs	2,922	7,168	40%
Logic Distribution			
Number of occupied Slices	2,395	3,584	66%
Number of Slices containing only related logic	2,395	2,395	100%

Number of Slices containing unrelated logic	0	2,395	0%
Total Number of 4 input LUTs	3,064	7,168	42%
Number used as logic	2,773		
Number used as a route-thru	142		
Number of bonded IOBs	2	195	1%
Number of BUFGMUXs	1	24	4%

D. Comparison of Experimental and Analytical Methods

Sr. No.	Parameter	Using Traceback scheme	Using Register Exchange scheme	Available in Spartan 3A XC3S400A FPGA
1	Number of Slices registers	133 (3 %)	2381 (66 %)	3564
2	Number of Slice Flip Flops	97 (1 %)	1898 (26 %)	7168
3	Number of 4 input LUTs	182 (1 %)	2906 (40 %)	7168
4	Number of bonded IOBs	14 (7 %)	2 (1 %)	195
5	Number of BRAMs	1 (5 %)	2 (10 %)	20
6	Number of GCLKs	2 (8%)	2 (8 %)	24

Hence 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

VI. 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 by utilizing both method and Synthesis result shows that Traceback method is more efficient in term of Chip Area Utilization so as will be Power Consumption in comparision with Register Exchanged 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

REFERENCES:

1. Iakovos Mavroidis, "FPGA Implementation of the Viterbi Decoder", University of California Berkeley, Dec. 1999.
2. Miloš Pilipovic, Marija Tadic, "FPGA Implementation of Soft Input Viterbi Decoder for CDMA2000 System", 16th Telecommunications forum TELFOR 2008.
3. Inyup Kang, Member IEEE and Alan N. Wilson (1998). "Low Power Viterbi Decoder for CDMA Mobile Terminal*", IEEE Journal of Solid State Circuits. IEEE. Vol 33. p.p. 473-481, 2010.*
4. Viterbi, A."Convolutional codes and their performance in communication systems "IEEE Trans. Commun. Technol" ,VOl .Com 19, no ,5, Oct.1971, pp.715-772, 2009.
5.] John G. Proakis (2001). "Digital Communication". McGraw Hill, Singapore. *pp 502-507, 471-475, 2010.*
6. Hema.S, Suresh Babu.V, Ramesh P, "FPGA Implementation of Viterbi Decoder", 6th WSEAS Int. Conf. on Electronics, February 2007.
7. A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," IEEE Trans. Inform. Theory, vol. IT-13, pp. 260-269, Apr. 1967.
8. John G. Proakis (2001). "Digital Communication". McGraw Hill, Singapore. pp 502-507, 471-475.
9. Viterbi, A."Convolutional codes and their performance in communication systems "'IEEE Trans. Commun. Technol ,VOl .Com 19, no ,5, Oct.1971, pp.715-772.
10. S.Haykan, Communication Systems, Wiley, 1994.
11. C. Arun, V. Rajamani, "Design and VLSI implementation of a Low Probability of Error Viterbi decoder", First International Conference on Emerging Trends in Engineering and Technology, IEEE, 2008.
12. Kelvin Yi-Tse Lai, "An Efficient Metric Normalization Architecture for High-speed Low- Power Viterbi Decoder," IEEE 2007.