FPGA IMPLEMENTATION OF HIGH SPEED AND LOW POWER HYBRID VITERBI DECODER USING VLSI

Aakanksha Mani[1], Mr. Nitin Kumar[2]
[1]M.Tech (ECE), [2]Asst. Prof. Dept. of ECE, GITM, KABLANA, JHAJJAR

ABSTRACT: Error-correcting convolution codes provide a proven mechanism to limit the effects of noise indigital data transmission. Convolution codes are employed to implement forward error correction(FEC) but the complexity of corresponding decoders increases exponentially with the constraint length K. Convolution Encoding with Viterbi decoding is a powerful FEC technique that is particularly suited to a channel in which the transmitted signal is corrupted mainly by Additive white Gaussian Noise. A Viterbi decoder uses the Viterbi algorithm for decoding a bit stream that has been encoded using Forward error correction based on a Convolutional code. The maximum likelihood detection of a digital stream is possible by Viterbi algorithm. In this paper, we present a Convolutional encoder and Viterbi decoder with a constraint length of 9 and code rate of 1/2. This is realized using VHDL. It is simulated and synthesized using Xilinx 13.1i.

Keywords: Convolutional Encoder, Viterbi Decoder, Verilog HDL, Viterbi Algorithm.

I. INTRODUCTION

Convolution codes are used in a variety of systems including today's popular wireless Standards (such as 802.11) and in satellite communications. Standard for CDMA (Code Division Multiple Access), employs convolutional coding. The Viterbi decoding algorithm was proposed and analyzed by Viterbi in 1967 [1]. It is widely used as a decoding technique for convolutional codes as well as the bit detection method in storage devices. Viterbi decoders currently find their use in more than one billion cell phones. The algorithm works by forming trellis structure, which is eventually traced back for decoding the received information. Convolutional encoding with Viterbi decoding is a powerful method for forward error correction. The Viterbi algorithm essentially performs maximum likelihood decoding. However, it reduces the computational complexity by using trellis structure. Figure1 shows the convolutional encoder and Viterbi decoder, which is used in the digital communication system. Here, X is the input data steam, which is given into the convolutional encoder and it produces the encoded data stream (Y). The encoded data stream (Y) is given to the channel in the presence of noise. Hence, it produces the noise added encoded data stream (R).Finally, data stream (R) is given to the Viterbi decoder that produces the estimated data stream (Z) applied at the input [5].

II. CONVOLUTION ENCODER

A convolutional encoder is a linear system. A binary convolutional encoder can be represented as a shift register. The outputs of the encoder are modulo 2 sums of the values in the certain register's cells. The input to the encoder is either the un encoded sequence (for non-recursive codes) or the un encoded sequence added with the values of some register's cells (for recursive codes). Convolutional codes can be systematic and nonsystematic. Convolutional codes are usually described using two parameters: the code rate and the constraint length. The code rate is expressed as a ratio of number of input symbols (k) into the channel encoder to the number of output symbols (n) by the channel encoder in a given cycle. Then, the code rate is expressed as,r = k/n bits/symbol. The constraint length (K) denotes the length of the convolutional encoder. Convolutional encoder increases the length of the message sequence by adding redundant bits in order to increase the likelihood of detecting the transmitted sequence even if errors have occurred during transmission. Figure 2 shows the convolutional encoder of constraint length (K) = 9 and code rate(r) = 1/2.

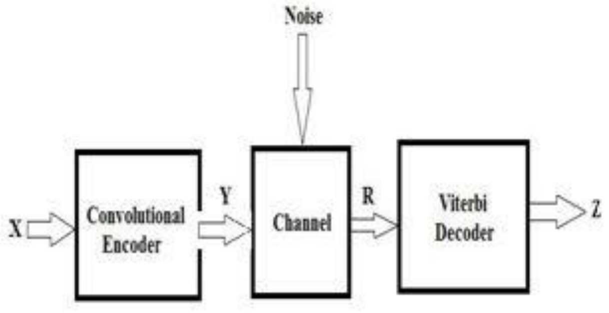


Figure 1: Convolutional Encoder and Viterbi Decoder

A convolutional encoder generates redundant bits by using modulo-2 convolutions. Hence, it is called as Convolutional encoder. If n modulo-2 adders are used, then it produces n outputs for each bit. The motivation of this paper is to realize a Viterbi decoder having Constraint length 9 and code rate 1/2 byusing Xilinx 13.1i tools. the state information of a convolutional encoder. In the state diagram, the state information of the convolutional encoder is shown in circles. Then, it is stored in the shift registers. Each new input information bit causes a transition from one state to another. The path information between the states has been denoted as x/c. Here, x represents the input information bits and c represents the output encoded bits. It is customary to begin convolutional encoding from the all zero state. For example, the input information sequence x={1011} (begin from the all zero state) leads to the state transition sequence s={10, 01, 10, 11}and produces the output encoded sequence c={11, 10, 00, 01}.

III. VITERBI ALGORITHM

Viterbi decoder is not a function of the number of symbols in the codeword sequence. The algorithm involves calculating a measure of similarity, or distance, between the received signal, at time ti, and all the trellis paths entering each state at time ti. The Viterbi algorithm removes from consideration those trellis paths that could not possibly be candidates for the maximum likelihood choice. When two paths enter the same state, the one having the best metric is chosen; this path is called the surviving path. This selection of surviving paths is performed for all the states. The decoder continues in this way to advance deeper into the trellis, making decisions by eliminating the least likely paths. In other words, it implies that the car first enter the stop state and then enter the Reverse state. Hence, when we receive the information through the processes of forward, reverse and stop, we can safely interpret it as – forward, stop and reverse as this is a “maximum likelihood sequence”. The Viterbi algorithm uses the trellis diagram to compute the path metric value (accumulated distance) from the received sequence to the possible transmitted sequences. The total number of such trellis paths increases exponentially with the number of stages in the trellis. It causes potential complexity and memory problems. The Viterbi decoding algorithm has been classified into hard decision decoding and soft decision decoding. If the received signal is converted into two levels, either zero or one, it is called hard decision. If the input signal is quantized and processed for more than two levels, it is called soft decision. The soft decision decoding is expensive and require large amount of memory than hard decision decoding. Hence, this work focuses on the hard decision decoding. Figure 4 shows the trellis diagram for hard decision Viterbi decoding. When a sequence of data is received from the channel, it is desirable to estimate the original sequence that has been sent. The process of identifying such a sequence can be done using a diagram called ‘trellis’. The detection of the original stream can be described as finding the most probable path through the trellis. In the trellis diagram each node specifies an individual state at a given time and indicates a possible pattern of recently received data bits. The transition to a new state at the next timing cycle is indicated by each branch [5].

IV. VITERBI DECODER

Viterbi algorithm is used in the Viterbi decoder for decoding abit stream that has been encoded using FEC based on a Convolutional code. Figure 5 shows the block diagram of Viterbi decoder [3]. It consists of the following functional units, namely, Branch Metric Unit, Path Metric Unit, Survivor Memory unit.

A. Branch Metric Unit: The comparison between received code symbol and expected code symbol is done by branch metric unit. It also counts the number of differing bits. It is the smallest unit in the Viterbi decoder. The measured value of the BMU can be the input decoding[4]. A branch metric unit’s function is to calculate branch metrics, which are formed distances between every possible symbol in the code alphabet, and the received symbol.

B. Path Metric Unit: The partial path metric of each branch is computed by the use oftwo adders. The partial path metric is compared by the comparator and an appropriate branch is selected by the selector. The selector selects the smaller value and stored that value as the new path metric for each state. The corresponding bit decision is transferred to the SMU [4].A path metric unit summarizes branch metrics to get metrics for 2K −1 paths, one of which can eventually be chosen as optimal. Every clock has some sequence it makes 2K −1 decisions, throwing off wittingly non optimal paths. The results of these decisions are written to the memory of a trace back unit.

C. Survivor Memory Unit: The Survivor Memory Unit receives the bit decision from the PMU. This will produces the decoded sequence. After finishing the SMU, it is important to perform the trace back module. When the trellis diagram is finished, the trace-back module will search the maximum likelihood path from the final state which is state0 to the beginning state which is state0.

V. IMPLEMENTATION

The Convolutional encoder and Viterbi decoder are implemented using VHDL and the code has been developed underfull-custom design. The Convolutional encoder encodes the given input sequence by modulo-2 adders. The Viterbi decoder decodes the original input sequence by using Viterbi algorithm From the trellis structure, we calculate the branch metric value and load that value. These values are added and compared. Then, the smallest value found in the process is the new path metric value. If all the states and the trellis stages are finished, then the trace back module is performed. It produces the decoded data (original input data).

VI. RESULTS AND DISCUSSION

The Viterbi decoder has been developed using and the synthesis is carried out. It has been simulated and the simulation result is shown in Figures.

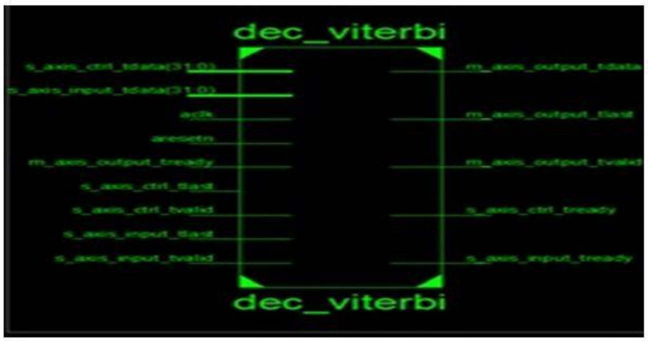


Figure 2. Top level RTL schematic view

Table 1. comparison among TB, RE, Hybrid

	Trace Back Method	Register Exchange Method	Existing Hybrid Method	Proposed Hybrid Method
Max. Frequenc y	343.525M HZ	158.983M HZ	74.914 MHZ	253.891 MHz

Max. Output Required Time after clock	35.829ns	23.829ns	9.338ns	3.921ns
Number of Slice Flip Flop	97	1910	446	3482

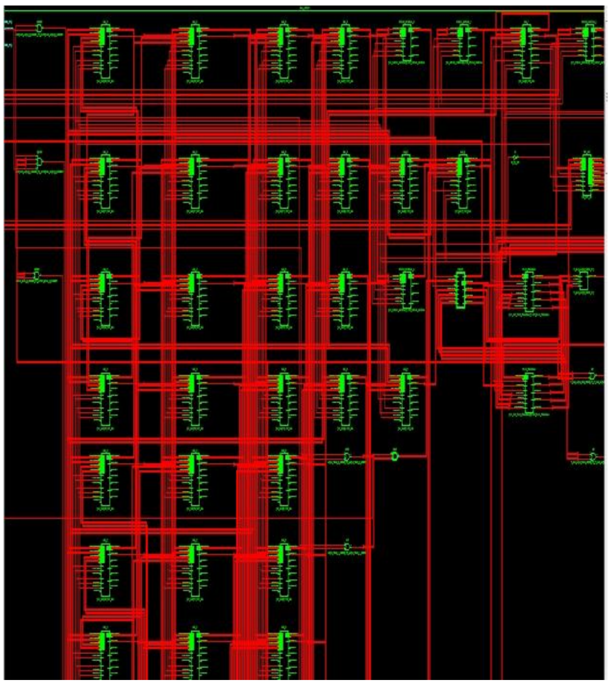
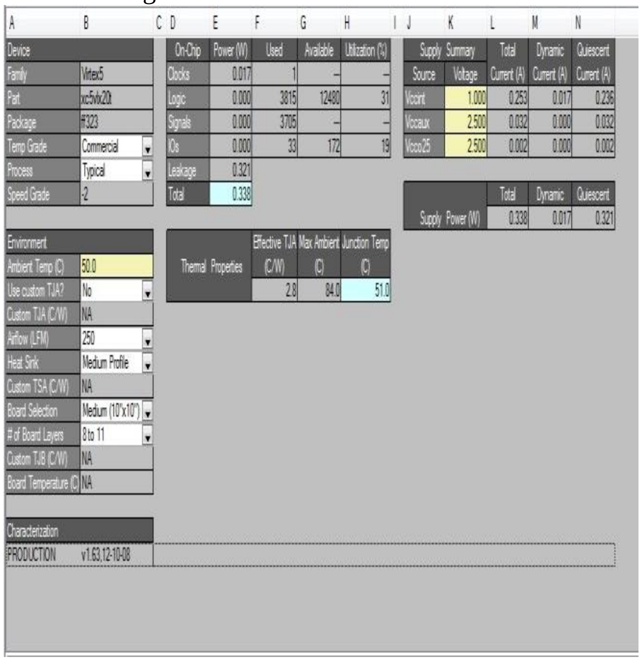


Figure 2. Internal RTL Schematic view



Device	
Family	Virtex5
Part	xc5vlx20t
Package	ff323
Temp Grade	Commercial
Process	Typical
Speed Grade	-2

On-Chip	Power (W)	Used	Available	Utilization (%)
Clocks	0.017	1	---	---
Logic	0.000	3815	12480	31
Signals	0.000	3705	---	---
IOs	0.000	33	172	19
Leakage	0.321			
Total	0.338			

Supply Summary		Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	1.000	0.253	0.017	0.236
Vccaux	2.500	0.032	0.000	0.032
Vcco25	2.500	0.002	0.000	0.002

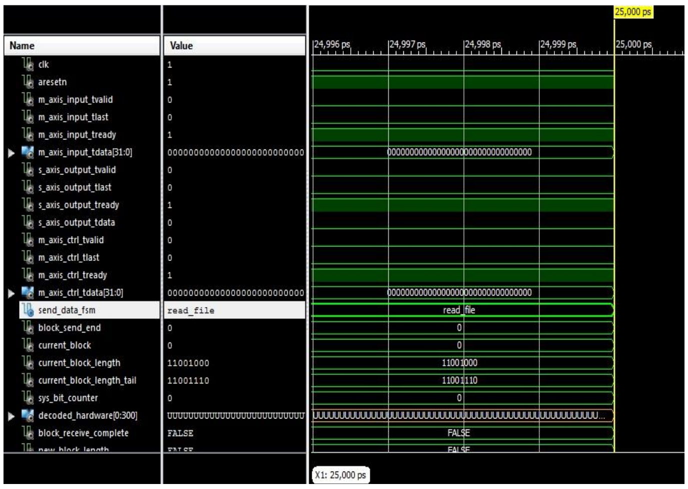
	Total	Dynamic	Quiescent
Supply Power (W)	0.338	0.017	0.321

Environment	
Ambient Temp (C)	50.0
Use custom TJA?	No
Custom TJA (C/W)	NA
Airflow (LFM)	250
Heat Sink	Medium Profile
Custom TSA (C/W)	NA
Board Selection	Medium (10"x10")
# of Board Layers	8 to 11
Custom TJB (C/W)	NA
Board Temperature (C)	NA

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	2.8	84.0	51.0

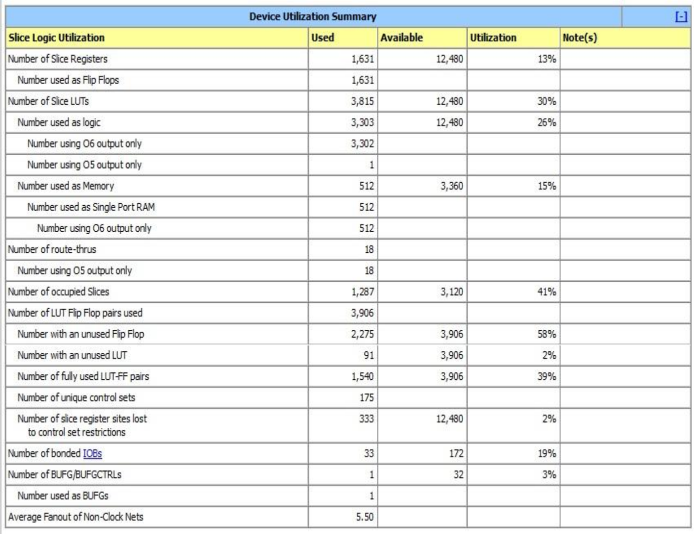
Characterization	
PRODUCTION	v1.63,12-10-08

Figure 3. Power distribution analysis



Name	Value
clk	1
aresetn	1
m_axis_input_tvalid	0
m_axis_input_tlast	0
m_axis_input_tready	1
m_axis_input_tdata[31:0]	00000000000000000000000000
s_axis_output_tvalid	0
s_axis_output_tlast	0
s_axis_output_tready	1
s_axis_output_tdata	0
m_axis_ctrl_tvalid	0
m_axis_ctrl_tlast	0
m_axis_ctrl_tready	1
m_axis_ctrl_tdata[31:0]	00000000000000000000000000
send_data_fsm	read_file
block_send_end	0
current_block	0
current_block_length	11001000
current_block_length_tail	11001110
sys_bit_counter	0
decoded_hardware[0:300]	UUUUUUUUUUUUUUUUUUUUUUUUUUUU
block_receive_complete	FALSE

Figure 4. Top level Simulation



Device Utilization Summary				
Slice Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Registers	1,631	12,480	13%	
Number used as Flip Flops	1,631			
Number of Slice LUTs	3,815	12,480	30%	
Number used as logic	3,303	12,480	26%	
Number using O6 output only	3,302			
Number using O5 output only	1			
Number used as Memory	512	3,360	15%	
Number used as Single Port RAM	512			
Number using O6 output only	512			
Number of route-thrus	18			
Number using O5 output only	18			
Number of occupied Slices	1,287	3,120	41%	
Number of LUT Flip Flop pairs used	3,906			
Number with an unused Flip Flop	2,275	3,906	58%	
Number with an unused LUT	91	3,906	2%	
Number of fully used LUT-FF pairs	1,540	3,906	39%	
Number of unique control sets	175			
Number of slice register sites lost to control set restrictions	333	12,480	2%	
Number of bonded IOBs	33	172	19%	
Number of BUFG/BUFGCTRLs	1	32	3%	
Number used as BUFGs	1			
Average Fanout of Non-Clock Nets	5.50			

Figure 5. Device utilization summary

VII. CONCLUSION

The use of error-correcting codes has proven to be an effective way to overcome data corruption in digital communication channels. Although widely-used, the most popular communications decoding algorithm, the Viterbi algorithm, requires an exponential increase in the hardware complexity to achieve grater decode accuracy. In this paper, we have presented the design and implementation of the Convolutional encoder and Viterbi decoder using VLSI Technique. This design has been simulated and synthesized using XILINX-ISE 13.1i forthe constraint length of K=9 and code rate of ½ input sequence. The given input sequence has been encoded by using convolution encoder and it is transmitted through the channel. Finally, the transmitted sequence is decoded by the Viterbi decoder and the estimated original sequence is produced.

REFERENCES

[1] A.J.Viterbi, "Error bounds for convolutional coding and an asymptotically optimum decoding algorithm", IEEE Tran.on Inform. Theory, Vol. 2, Pp. 260-269, Apr. 1967.

[2] Wong, Y.S. et.al "Implementation of convolutional encoder and Viterbi decoder using VHDL", IEEE Tran.on Inform. Theory, Pp. 22-25, Nov. 2009.

[3] Hema.S, Suresh Babu.V and amesh.P, "FPGA Implementation of Viterbi decoder", proceedings of the 6th WSEAS ICEHWOC, Feb. 2007.

[4] IrfanHabib, ÖzgünPaker, Sergei Sawitzki, "Design Space Exploration of Hard-Decision Viterbi Decoding: Algorithm and VLSI Implementation", IEEE Tran.on Very Large Scale Integration (VLSI) Systems, Vol. 18, Pp. 794-807, May 2010.

[5] G. Madhu Kumar, A. Swetha, "Design and implementation of convolution encoder and viterbi decoder", Volume: 1, Issue: 6, Nov 2012, ISSN No 2277 – 8179.

[6] Dr. Vinodkumar Jacob, Dr. M. Bhasi and Dr. Gopikakumari, "Characterization of Measurement Error and Uncertainty in Working Standards", International Journal of Advanced Research in Engineering & Technology (IJARET), Volume 4, Issue 5, 2013, pp. 106 - 125, ISSN Print: 0976-6480, ISSN Online: 0976-6499

[7] Chaiwat Keawsai, Keattisak Sripimanwat, Attasit Lasakul. "Modified register exchange method of Viterbi decoder for 3GPP Mobile System".

[8] Yun-Nan Chang, Hiroshi Suzuki, and Keshab K. Parhi (2000). "A 2-Mb/s 256-State 10-mW Rate-1/3 Viterbi Decoder". IEEE Journal of Solid-State Circuits. Vol. 35 pp.826-833.

[9] Christian Schuler and GMD IOKH. "Code Generation Tools for hardware implementation of FEC Circuits".

[10] M.kivioja, J.isoaho and L.vanska (1999). "Design and Implementation of Viterbi Decoder with FPGAs". Journal of VLSI Signal Processing.pp.5-14

[11] John Davis, Andrew Lin, Njuguna Njoroge, Ayodele Thomas (2002). "The Viterbi algorithm as stream application".