FPGA Implementation of 413.121 MHz and 11.34 mW High Speed Low Power Viterbi Decoder

Pooran Singh and Santosh Kr. Vishvakarma

***Abstract*—High speed and low power Viterbi Decoder of rate ½ convolutional coding with a constraint length K = 3 is presented in this paper. After implementation of proposed Viterbi decoder in Virtex 7 Field Programmable Gate Array (FPGA) kit we come to know that it's functioning on 413.121 MHz clock and in such a high speed it also maintain a low power of 11.34 mW in Spartan 6 FPGA. Since the FPGA boards used are different and from that we justified that using both logics together in one Integrated Circuit (IC) we can create a high speed and low power Viterbi decoder at the same time with some extra hardware area.**

***Index Terms*—FPGA, viterbi decoder, low power, xilinx power estimator, high speed.**

I. Introduction

Convolutional (Viterbi) decoding is a Forward Error Correction (FEC) technique [1]. The purpose of FEC is to improve the capacity of a channel by adding some carefully designed redundant information to the data being transmitted through the channel. 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.

The Viterbi decoding algorithm, proposed in 1967 by Viterbi [2], is a decoding process for convolutional codes in memory less noise channel. The algorithm can be applied to a host of problems encountered in the design of communication systems. The Viterbi decoding algorithm provides a maximum-likelihood algorithm. A maximum likelihood algorithm identifies a code word that maximizes the conditional probability of the received code word against the decoded code word. The algorithm gives the same results when the source information has a uniform distribution.

In this paper, a design of high speed low power Viterbi decoder at the RTL level in the standard cell design environment is proposed. In the standard cell design environment, the behavior of a design is described in VHDL. The behavioral design is synthesized to generate a gate level design. The gate-level design is placed and routed to generate a layout of the design. The advantages of a standard cell based design over full custom design are faster turnaround time for the design, ease in design verification and more accurate modeling of the circuit. Design of Viterbi decoders at the RTL-level is focused here. Viterbi algorithms and implementation of Viterbi decoders were investigated intensively in the past three decades [3]-[5].

This paper is organized as follows. The background on the operation of Viterbi algorithm is provided in Section II. Section III describes the design for Viterbi decoders. Section IV shows the power and speed in various FPGA kits. Section V summarizes the paper.

Manuscript received November 2, 2012; revised January 24, 2013.

The authors are with Electrical Department, Digital System Design Lab, IIT Indore, MP, India (e-mail: phd11120203@iiti.ac.in, skvishvakarma@iiti.ac.in).

II. Viterbi Decoding Algorithm

Viterbi decoding is one of two types of decoding algorithms [6], [7] used with convolutional encoding the other type is sequential decoding. Sequential decoding has the advantage that it can perform very well with long-constraint-length convolutional codes, but it has a variable decoding time. A discussion of sequential decoding algorithms is beyond the scope of this paper.

Viterbi decoding has the advantage that it has a fixed decoding time. It is well suited to hardware decoder implementation. But its computational requirements grow exponentially as a function of the constraint length, so it is usually limited in practice to constraint lengths of $K = 9$ or less. Stanford Telecom produces a $K = 9$ Viterbi decoder that operates at rates up to 96 kbps, and a $K = 7$ Viterbi decoder that operates at up to 45 Mbps. Advanced Wireless Technologies (AWT) offers a $K = 9$ Viterbi decoder that operates at rates up to 2 Mbps.

For years, convolutional coding with Viterbi decoding has been the predominant FEC technique used in space communications, particularly in geostationary satellite communication networks, such as VSAT (very small aperture terminal) networks. I believe the most common variant used in VSAT networks is rate 1/2 convolutional coding using a code with a constraint length $K = 7$. With this code, you can transmit binary or quaternary phase-shift-keyed (BPSK or QPSK) signals with at least 5 dB less power than you'd need without it. That is a reduction in watts of more than a factor of three. This is very useful for reducing transmitter and/or antenna cost or permitting increased data rates given the same transmitter power and antenna sizes. Many radio channels are AWGN channels, but many, particularly terrestrial radio channels also have other impairments, such as multipath, selective fading, interference, and atmospheric (lightning) noise. Transmitters

DOI: 10.7763/IJMO.2013.V3.226

and receivers can add spurious signals and phase noise to the desired signal as well. Although convolutional coding with Viterbi decoding might be useful in dealing with those other problems, it may not be the most optimal technique.

III. Implementation of Viterbi Decoder

The major tasks in the Viterbi decoding process are as follows:

- Branch metric computation.
- State metric update: Update the state metric using the new branch metric.
- Survivor path recording: Tag the surviving path at each node.
- Output decision generation: Generation of the decoded output sequence based on the survivor path information.

Fig. 1 shows the proposed Viterbi decoder. This section discusses the different parts of the Viterbi decoding process. Analog signals are quantized and converted into digital signals in the quantization block. The synchronization block detects the frame boundaries of code words and symbol boundaries. We assume that a Viterbi decoder receives parallel successive code symbols, in which the boundaries of the symbols and the frames have been identified.

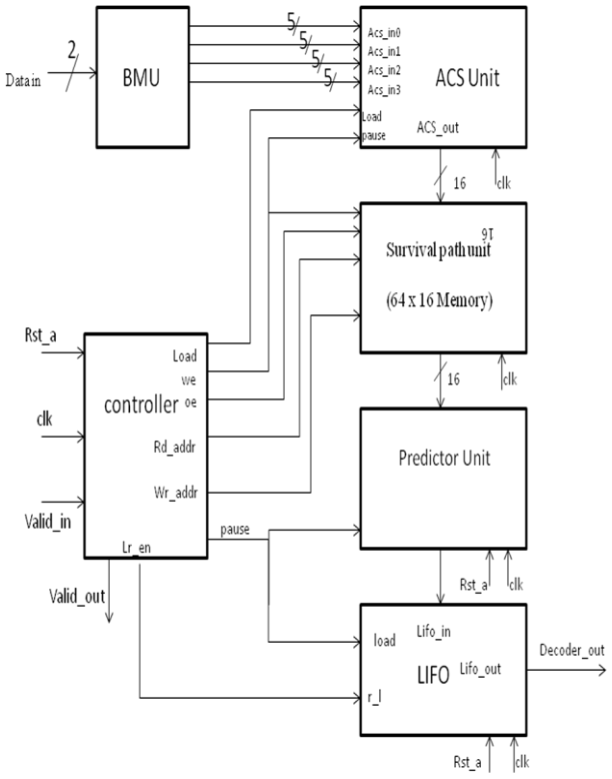

Fig. 1. Internal sub blocks of proposed Viterbi decoder.

A. Branch Matrix Unit

It is used to generate branch metrics, which are hamming distances of input data from 00, 01, 10 and 11. The BM unit is used to calculate branch metric for all trellis branches from the input data. We choose absolute difference as measure for branch metric. These branch metrics are viewed as being the weights of the branches. The block diagram of BMU is shown in Fig. 2.

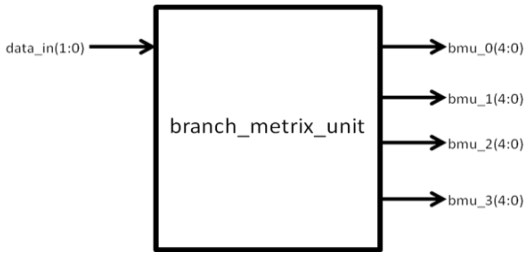

Fig. 2. Block diagram of branch matrix unit.

B. ACS (Add Compare Select) Unit

A new value of the state metrics has to be computed at each time instant. In other words, the state metrics have to be updated every clock cycle. Because of this recursion, pipelining, a common approach to increase the throughput of the system, is not applicable. The Add-Compare-Select (ACS) unit hence is the module that consumes the most power and area. Fig. 3 shows the internal block diagram of single ACS sub block.

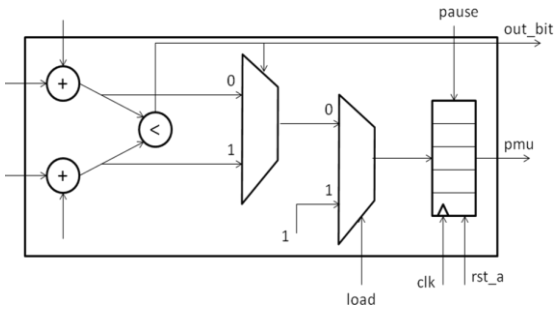

Fig. 3. Internal diagram of a single ACS sub block.

In order to obtain the required precision, a resolution of 5 bits for the state metrics is essential, while 5 bits are needed for the branch metrics. Since the state metrics are always positive numbers and since only positive branch metrics are added to them, the accumulated metrics would grow indefinitely without normalization. The operation of the ACS unit is shown in Fig. 4. The new branch metrics are added to previous state metrics to form the candidates for the new state metrics. The comparison can be done by using the subtraction of the two candidate state metrics, and the MSB of the difference points to a larger one of two.

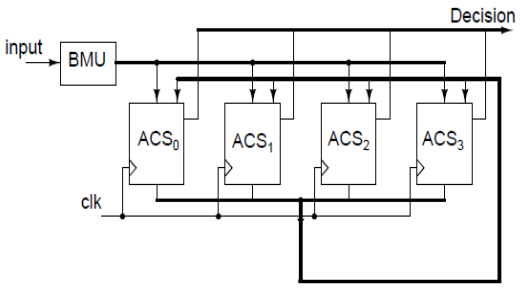

Fig. 4. ACS unit.

C. Memory

Memory is required to store the survivor Path Matrix Unit (PMU) [8]. The word length of the memory depends on the number of the ACS sub-blocks used in the design or the total number of states in the decoder or *k*^2 (where k is the

constraint length, 5 in our case), and the depth of the memory depends on the trellis length. The memory depth usually should be kept two times the trellis length or two blocks of memory equal to trellis length. We have for our project k = 5 and trellis length equal to 32, so the memory block used is 64x16. The memory used is dual port. One port for writing the data and other for reading the data, as we need to write and read the data simultaneously and that to from different addresses. Memory should write data synchronously but the reading of the data should be asynchronous to keep the latency low or better manage the synchronous behavior of the full system.

D. Controller

A controller is used to synchronize between the different modules of the system. The controller unit of decoder controls signals like we, pause, valid_out, oe, rd_addr, wr_addr etc.*we, oe* => write enable and output enable for the memory; *wr_addr* => write address of the memory. *wr_addr* is assigned the output of a 6 bit counter which counts up from "000000" to "111111"; *rd_addr* => read address of the memory. *rd_addr* is assigned the output of another 6 bit counter which counts down from "111111" to "000000"; *pause* => *pause* signal works as an enable signal for other synchronous modules. When it goes high then the modules stops for a moment and again starts functioning when the pause signal goes low; *valid_out* => *valid_out* signal is asserted when the decoder starts providing valid decoded data at the output. It goes low every time the pause signal goes high; *lr_en*=> the *lr_en* signal controls the read, write and left shift, right shift function of the LIFO.The controller also includes two six bit counters of which one counts up another counts down. These counters drive the write and read addresses of the memory. The pause signal generated by the controller also stops these counters for a while so that no unnecessary data is written onto the memory or read from it.

E. Predictor Unit

Predictor unit is used to trace back the trellis sequence of length 32 and predict the next state and actual decoded bit after rectifying the error. This unit is a state machine that is loaded with the state with minimum accumulated path metric after every 32 clock cycles. This unit uses the state value to access a bit from the path metric memory unit (PMM) or memory unit. Block diagram of predictor unit is shown Fig. 5.

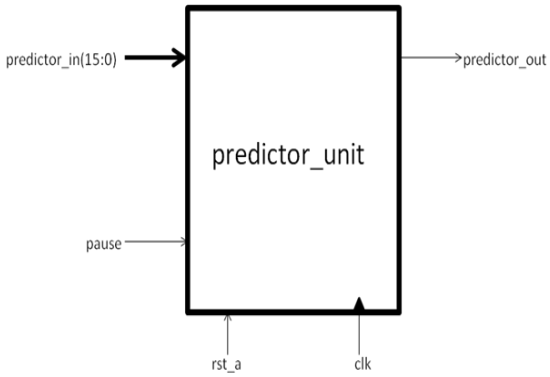

Fig. 5. Block diagram of predictor unit.

Pin description of predictor unit is shown in Table I.

TABLE I: PIN DESCRIPTION OF PREDICTOR UNIT

If rst_a is high	Fsm become at state s0(initial state)
If pause is high	Fsm remain in the same state and output remain in the same value
Other conditions	State transition occurs in fsm and generate the output

Functional description of predictor unit is in Table II.

TABLE II: FUNCTIONAL DESCRIPTION OF PREDICTOR UNIT

S.No.	Pin name	Width	Direction	Description
1	Predictor_in	16	Input	Input data
2	Predictor_out	1	Output	Output data
3	rst_a	1	Input	Input data
4	Clk	1	Input	Clock signal
5	Pause	1	Input	Input data

Predictor unit consist of 16:1 multiplexer and a state machine which is used to generate as well as rectify the error. Each present state of state machine has only two possible next states. According to the present state of the state machine, respective multiplexer input is selected and then check that bit, if the bit is 0 then the smaller possible state is selected and if the bit 1 then bigger state is selected. Based on the present state and next state data, actual decoded output is generated which is then send to the LIFO input. Internal circuit of the predictor unit is shown in Fig. 6 and it's FSM in Fig. 7.

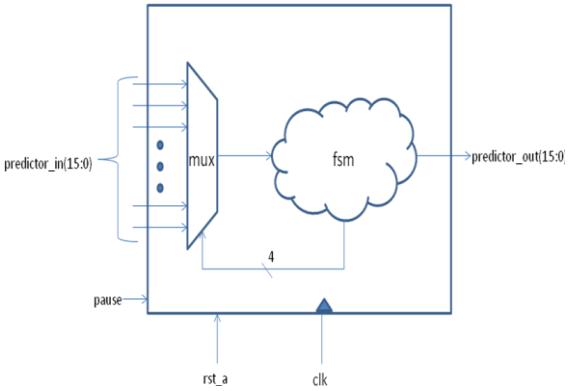

Fig. 6. Internal block description of predictor unit.

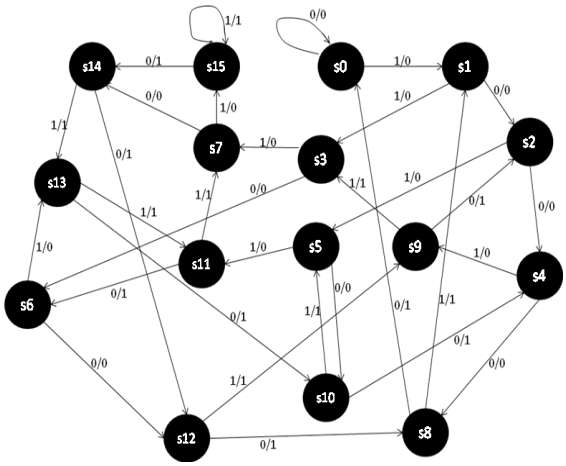

Fig. 7. FSM of predictor unit.

F. LIFO Unit

Every 32 decoded bits put out by the predictor unit is in reverse order of the transmitted data, this necessitates a LIFO unit. This unit has 2 32-bit registers in which one of the register is read while the other is written. The two 32-bit registers are read and write alternatively and simultaneously selected by the multiplexer which is in the read mode and gives the output in correct sequence. The block diagram of LIFO is shown in Fig. 8.

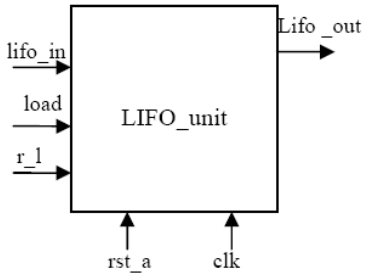

Fig. 8. Block diagram of LIFO unit

Pin description of LIFO unit is shown in Table III.

TABLE III: PIN DESCRIPTION OF LIFO UNIT

S.No.	pin name	width	direction	description
1	lifo_in	1	Input	Input signal
2	lifo_out	1	Output	Output signal
3	rst_a	1	Input	Input Signal
4	clk	1	Input	Clock signal
5	load	1	Input	Input signal
6	r_l	1	Input	Input signal

Functional description of LIFO unit is shown in Table IV.

TABLE IV: FUNCTIONAL DESCRIPTION OF LIFO UNIT

If rst_a is high	Both 32-bit registers become reset
If load is high	Read and write the data from the registers
If load is low	Neither read nor write the data
If r_l is high	Upper register in read mode and lower in write mode
If r_l is low	Upper register in write mode and lower in read mode

IV. RESULTS

The implementation of Viterbi decoder with K=3 is done. Our work is based on about to make a Viterbi Decoder fast with low power consumption. So for getting high speed we used Virtex 7 Kit to generate a 413.121 MHz clock frequency of our proposed decoder and for low power optimization we used Spartan 6 Low voltage Xilinx FPGA [9] which shows a power reduction of Viterbi decoder upto 11.34 mW. The results is calculated in Xilinx synthesis tool and the power optimization is done in X power Estimator in Xilinx. The results shows that there is about 11.34 mW of Dynamic Power consumed by the logic and it run with 413.121 MHz Clock frequency.

A. Synthesis Report

In following synthesis report, Table V shows the logic utilization and Table VI shows the clock information while Table VII shows as the minimum power consumption of Spartan 6 low voltage FPGA kit.

1) Device used for high speed of 413.121 MHz

Target Device: Vertex7, Family: XC7VX330T, Package: FFG1157, Speed: -3.

TABLE V: LOGIC UTILIZATION INFORMATION

Slice Logic	No. of Logic used
No. of Slice Registers	180 out of 408000
No. of Slice LUTs	382 out of 204000
No. used as Logic	360 out of 204000
No. used as Memory	22 out of 70200
No. used as RAM	22
Number of 6 input LUT Flip Flop pairs used	392

TABLE VI: CLOCK INFORMATION

Minimum period	2.421ns(Maximum Frequency: **413.121MHz**)
Setup time	2.103ns
Hold time	1.106ns

2) Device used for 11. 34 mW low power.

Target Device: Spartan 6 low Power, Family: XC6SLX4L, Package: TQG144, Speed: -1L

TABLE VII: POWER SUPPLY SUMMARY

Supply Power (mW)	Dynamic	Quiescent	Total
	11.34	0.37	**11.34**

V. CONCLUSION

Hence from our proposed Viterbi decoder we designed a high speed of 413.121 MHz clock and low power with 11.34 mW total power consumption decoder and which can be used for communication protocols like CDMA for high speed data transmission with nearly low power with less amount extra logic utilization.

REFERENCES

[1] V. Tomas, "Decoding of convolutional codes over the erasure channel," *IEEE Trans. on Information Theory*, vol. 58, no. 1, pp. 90-108, Jan. 2012.

[2] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, April 1967.

[3] Bupesh Pandita and Subir K Roy, "Design and Implementation of viterbi decoder using FPGAs," *VLSID '99 Proceedings of the 12th International Conference on VLSI Design, page 611*, 1999.

[4] P. J. Black and T. H. Meng, "A 1-Gb/s, four-state, sliding block Viterbi decoder," *IEEE Journal of Solid-State Circuits*, vol. 32 no. 6, pp.797–805, 1997.

[5] M. Boo, F. Arguello, J. D. Bruguera, R. Doallo, and E. L. Zapata., "High-performance VLSI architecture for the Viterbi algorithm," *IEEE Trans. on communications*, vol. 45, no. 2, pp.168–176, 1997.

[6] O. Collins and F. Pollara., "Memory management in traceback Viterbi decoders," *TDA Prog. Rep*, pp. 42–99, 1989.

[7] G. Fettweis and H. Meyr., "Parallel Viterbi decoding by breaking the compare select feedback bottleneck," *Communications*, vol. 201, no. 88, 1988.

[8] G. Feygin and P. Gulak., "Architectural tradeoffs for survivor sequence memory management in Viterbi decoders," *IEEE Transactions on Communications*, vol. 41, no. 3, pp. 425– 429, 1993.

[9] Xilinx, "Lowering power at 28 nm with Xilinx 7 Series FPGAs," *White paper, WP389 (v1.1.1)*, 2012.

Pooran Singh is pursing his Ph.D. from IIT, Indore and he is in the process of developing the various power reduction techniques in FPGA at RTL level, Circuit level and by introducing the novel MOS devices in some of the module in FPGA. He received his M.Tech from ABV-IIITM, Gwalior in 2010.

Santosh Kumar Vishvakarma received his Bachelor of Science (B.Sc.) in Electronics from University of Gorakhpur, Uttar Pradesh India, Master of Science (M.Sc.) in Electronics from D.A.V.V. Indore, Madhya Pradesh, India and Master of Technology (M. Tech.) in Microelectronics from Punjab University, Chandigarh, India in 1999, 2001 and 2003 respectively. He has awarded his Ph.D. in Feb 2010 from IIT, Roorkee, India. At present Dr. Vishvakarma is with Electrical Engineering, School of Engineering, Indian Institute of Technology (IIT), Indore, India where he is leading Nanoscale Devices and VLSI/ULSI Circuit and System Design Lab. His present areas of research are Nanoscale Devices and Circuits, Ultra Low Power Digital & Analog Circuit Design and their Technology, Multigate and Multifin MOSFET (Square, Circular and Rectangular Gate All Around (GAA) MOSFET, Double Gate (DG) MOSFET, FinFET etc.) and their Circuit Applications in Memories.