An Efficient Low Power Convolutional Coding with Viterbi Decoding using FSM

S.Nanthini Devi[1], S.Devipriya[2], V.Maharani[3], S.Aarthy[4] and S.Gnanamurugan[5]

[1]*UG Scholar, Department of ECE, Vivekanandha College of Engineering for Women, India. Email: snanthini26@gmail.com*
[2]*UG Scholar, Department of ECE, Vivekanandha College of Engineering for Women, India. Email: divashna11@gmail.com*
[3]*UG Scholar, Department of ECE, Vivekanandha College of Engineering for Women, India. Email: mahahoney1996@gmail.com*
[4]*UG Scholar, Department of ECE, Vivekanandha College of Engineering for Women, India. Email: aarthy130896@gmail.com*
[5]*Assistant Professor, Department of ECE, Vivekanandha College of Engineering for Women, India. Email: sgm@gmail.com*

Article Received: 25 February 2017 *Article Accepted: 08 March 2017* *Article Published: 11 March 2017*

ABSTRACT

For wireless communication, the demand for high speed, low power and low cost Viterbi decoding are always required. Convolutional coding with Viterbi decoding is a very powerful method for forward error correction and detection method. It has been popularly used in many wireless communication systems to improve the limited capacity of its communication channels. VLSI technology in advance is using low power, less area and high speed constraints is often used for encoding and decoding of data.

Keywords: Convolutional coding, Viterbi decoding, Forward error correction and detection, Low power, Wireless communication systems and Channel encoding.

1. INTRODUCTION

Convolutional encoding is the most popularly implemented type of forward error correction (FEC) encoding method in wireless communication system. The modern complexity and a very good performance have made this Viterbi decoding as the preferred decoding method for convolutional codes to overcome all those transmission errors. Viterbi decoding is an optimal decoding method for any convolutional coding that is applied to additive white Gaussian noise channel during transmission. The Viterbi decoding algorithm occupies a huge memory, and very good computational resources along with high power consumption to address all those issues. With the upcoming use of digital communication, there is an increased interest in high-speed Viterbi decoder design within a single chip.

2. METHODOLOGY

Viterbi decoder design specification can be formulated by using application and literature review. All the other modules of this Viterbi Decoder block are to be identified. All the other sub-modules of the Adaptive Viterbi Decoder block are to be used in Verilog HDL and to be simulated using ModelSim. A test bench for verifying the Viterbi Decoder block completely is by developing all those in Verilog HDL using ModelSim. Design synthesis can be performed by using Xilinx ISE.

3. SIGNIFICANCE

Error correction and detection probability might be possibly reduced by transmitting more number of bits than needed to represent all the information's being sent, and for convoluting each bit with neighboring bits, there will be at least one bit that will be corrupted and enough information can be carried through by the neighboring bits for estimating what are the corrupted bits. This approach of transforming the number of information bits into a larger number of transmitted bits is called as channel encoding, and the particular approach of convolving of bits to distribute this information is referred to as the convolutional encoding. Convolutional encoding is very frequently used in noisy channels to correct all the errors. They would rather have good correcting capability or they perform very well even on very bad channels. Convolutional encoding is widely used in satellite communications. Even though convolutional coding is a very simple procedure, decoding of this convolutional coding is much more complex task than anything else. We use Viterbi decoder to overcome the drawbacks.

4. CONVOLUTIONAL CODING

A convolutional code is a type of error-correction coding method that generates parity symbols via the sliding application of a Boolean polynomial function to the data stream. The ability to perform the economical maximum likelihood soft decision decoding is one of the major benefits of the convolutional coding. This is in contrast to the classic block codes, which is generally represented by a time-variant trellis and therefore is typically hard-decision decoding.

The base coding rate is typically given by n/k, where n is input data rate and k is output symbol rate. The depth are often called as the "constraint length" 'K', where the output is a function of current input as well as previous (K-1) inputs.

Convolutional codes are often described as continuous. However, it may also be said that the convolutional coding have arbitrary block length, rather than being a continuous, since most of the real-world convolutional encoding is performed on the blocks of data. Convolutionally encoded block codes are typically employed in termination. The arbitrary block length of the convolutional coding can be contrasted into a classic block codes, which generally have a fixed block length that is determined by algebraic properties.

The code rate of the convolutional coding is commonly modified via symbol puncturing. For example, this convolutional coding with a 'mother' code rate n/k=1/2 can be punctured to a higher rate, for example, 7/8 simply not by transmitting a part of code symbols. The performance of punctured convolutional coding generally scales well with all the amount of parity transmitted. The ability to perform the economically soft decision decoding on convolutional coding, as well as the block length and code rate of convolutional coding, makes them a very popular one for digital communications.

5. VITERBI DECODER

The receiver can deliver hard or soft symbols to Viterbi decoder. A hard symbol is equivalent to the binary +/-1. A soft symbolism multileveled to represent the confidence in the bit being positive or negative. In case of the hard decision demodulation, data can be demodulated into either 1s or 0s, or may be quantized into two levels only. The process described above can make a hard binary decision about each of the incoming bits and then uses only its Hamming distances. This simplifiers the hardware, but does not result in optimal performance.

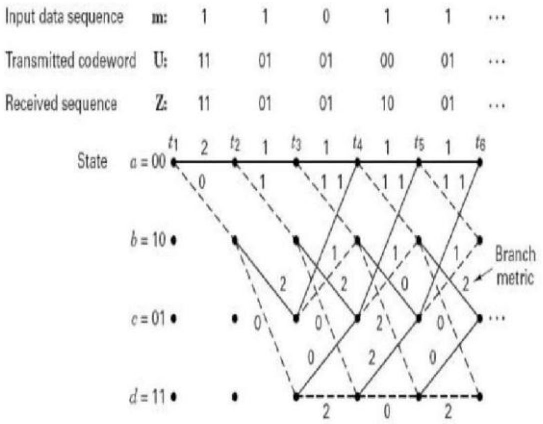

Input data sequence m: 1 1 0 1 1 ...
Transmitted codeword U: 11 01 01 00 01 ...
Received sequence Z: 11 01 01 10 01 ...
State
a = 00
b = 10
c = 01
d = 11
t1
t2
t3
t4
t5
t6
Branch metric
...

Fig.1. Block diagram of Viterbi decoder

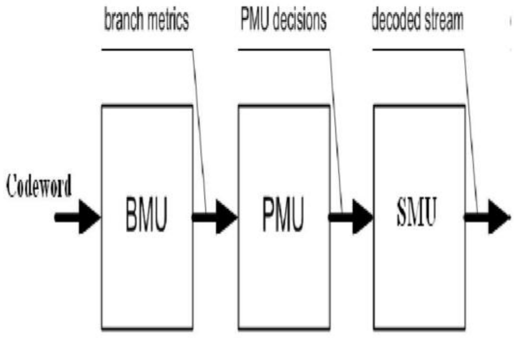


Fig.2. Trellis diagram of Viterbi decoder

Branch metric and path metric can be calculated by using hamming code. Codeword is given to branch metric unit and the Branch metric unit's function is to calculate all the branch metrics, which are hamming distances between every possible symbol in the Codeword and the received symbol. Path metric unit explains branch metrics to get the metrics for the 2K − 1 path, one of which can eventually be chosen as the optimal.

Survivor memory unit can be a trace-back process or a register exchange method, where this survivor path and this output data are identified. The error probabilities can be achieved by Viterbi algorithm which depends on the code and the rate of the code along with its free distance, channel SNR and demodulation Quantized output.

5.1 Adaptive Viterbi Algorithm

The adaptive Viterbi algorithm only keeps a number of most likely states instead of whole of 2k-1 state, where there is constraint length of convolution encoder. The rest of all the other states are discarded. The selection is based on likelihood or metric value of the path, which for hard decision is the hamming distance and a soft decision decoder is Euclidean distance.

5.2 Adaptive Viterbi Decoder (AVD)

Figure shows the data flow diagram of the adaptive Viterbi algorithm, which adds two functional blocks which includes the best winner search and the non-survivor purge, into the original Viterbi algorithm.

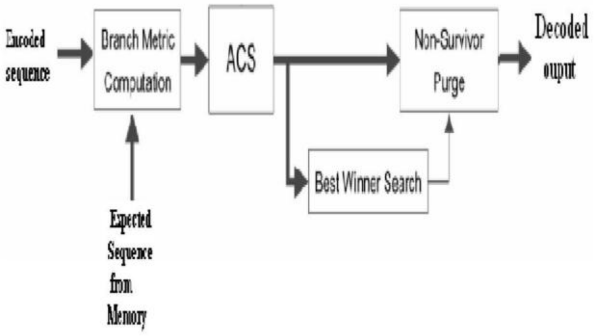


Fig.3. Block diagram of Adaptive Viterbi decoder

Codeword can be applied to the branch metric computation unit which calculates branch metric by comparing it with the expected symbol. It updates path metric by cumulative accumulation of branch metric unit. Best winner search determines final winner and gives it the non-survivor purge unit. It deletes all the paths expect winner. The first unit is called as branch metric unit which is the simplest block in the Viterbi decoder design. Here the received data symbols are compared to ideal outputs of encoder from the transmitter and branch metric unit is calculated. Hamming distance is used for branch metric unit computation.

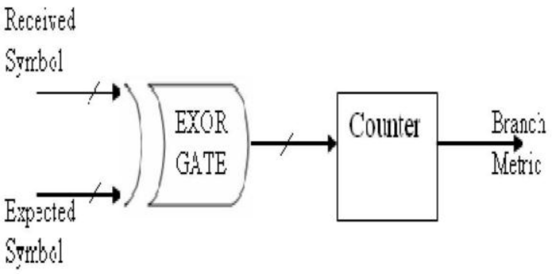


Fig.4. Block diagram of Branch Metric Unit

Block diagram of Branch metric unit is shown in Figure. The branch metric unit calculates the branch metrics from the input data. Hamming distance between the received

Codeword and the expected code word is calculated by comparing the received code symbol with the expected code symbol and by counting the number of different bits. The major task of the add compare select is to calculate the metrics and selected paths.

The add-compare-select unit recursively accumulates many branch metrics to path metrics for all the incoming paths of each and every state and selects the path with minimum path metric as the survivor path.

An add control select module is shown in Figure. The two adders compute the partial path metric unit of each branch, the comparator compares the two partial branch and path metrics, and the selector selects an appropriate branch.

Add compare select units determine their own local winners and the best winner search block finds the one which having the best minimum path metric among all the winners.

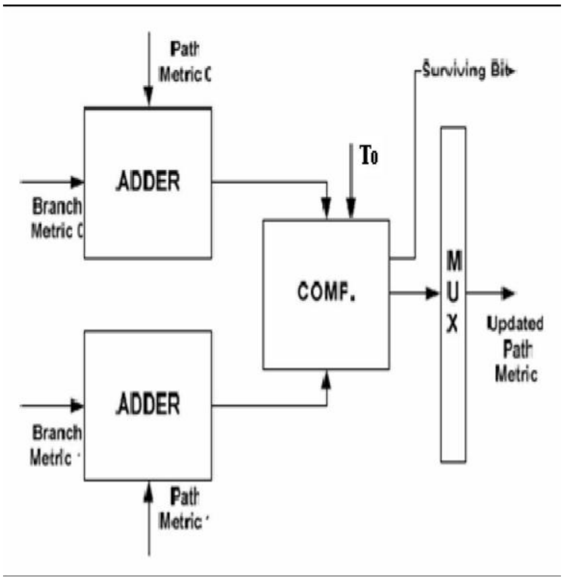


Fig.5. Add-Compare-Select Unit

Survivor Unit of the adaptive Viterbi decoder is based on either a Register Exchange or Trace-Back design style. The trace-back method takes up less area but requires much more time as compared to register exchange method because it needs to search or trace the survivor path back sequentially. The trace back approach might be the lower power alternative to the register exchange method. Block diagram of Trace back unit is shown in Figure. In the trace-back method, the storage can be implemented as RAM and is called the path memory. Comparisons in the add compare select unit and not the actual survivors are stored. After at least branches have been processed, the trellis connections are recalled in the reverse order and the path is traced back through the trellis diagram. The trace back method extracts the decoded bits, beginning from the state with the minimum Path Metric.

The Register Exchange method is the very simplest conceptually and a commonly used technique. The data path in Register Exchange is shown in For instance, at time slot T1 the survivor branch for state 1 is from state 0 at T0; therefore, the initial content of the state 0 register, which is a 0 is shifted into state 1 register at T1 and the corresponding decoded data for the survivor branch, which is a 1 is appended to it. Bold arrow indicate Global winner path. This method eliminates all the needs to trace back from the register of the final state contains the decoded output sequences. However, this method results in a complex hardware description structure due to the need to copy the contents from the registers in one stage to the other stage.

6. PROGRAMMABLE DEVICES

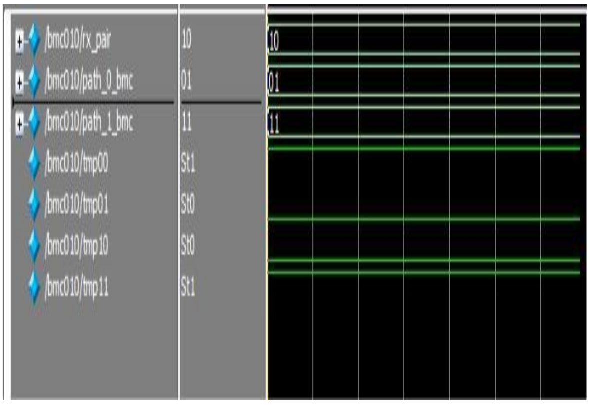
Programmable devices are those devices which can be programmed by the user. Various programmable devices are PLDs, CPLDs, ASICs and FPGAs. A *Field Programmable Gate Arrays* Field Programmable' means that the FPGA's function is defined by a user's program rather than by the manufacturer of the device.

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 maybe simple flip-flops or more complete blocks of memories [1-4].

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 takeover 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.

7. SIMULATION RESULTS

BRANCH METRIC UNIT STATE– 10(0)



```
/bmc010/rx_pair       10    10
/bmc010/path_0_bmc    01    01
/bmc010/path_1_bmc    11    11
/bmc010/tmp00         St1
/bmc010/tmp01         St0
/bmc010/tmp10         St0
/bmc010/tmp11         St1
```

Fig.6. BMU State 0

BMU at the state 10 Path_0 (01), Path_1 (11), BMP 00-1, 01-0, 10-0, 11-1.

BRANCH METRIC UNIT STATE-11(1)

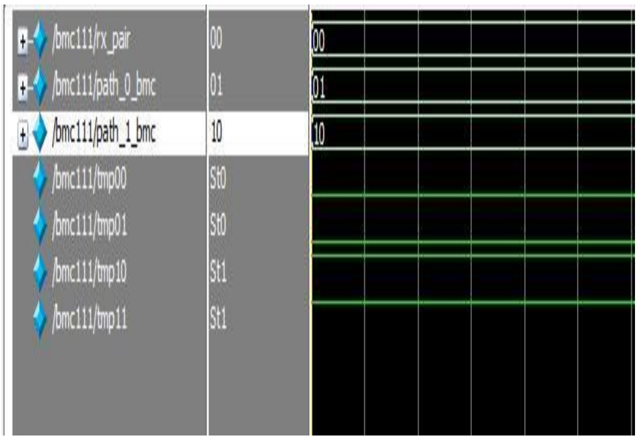

/bmc111/rx_pair 00
/bmc111/path_0_bmc 01
/bmc111/path_1_bmc 10
/bmc111/tmp00 St0
/bmc111/tmp01 St0
/bmc111/tmp10 St1
/bmc111/tmp11 St1

Fig.7. BMU State 1

BMU at the state 11 Path_0 (01), Path_1 (10), BMP 00-0, 01-0, 10-1, 11-1.

ADD-COMPARE SELECT UNIT

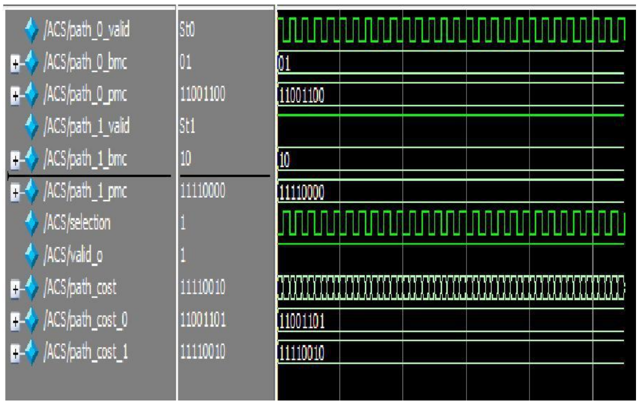

/ACS/path_0_valid St0
/ACS/path_0_bmc 01
/ACS/path_0_pmc 11001100
/ACS/path_1_valid St1
/ACS/path_1_bmc 10
/ACS/path_1_pmc 11110000
/ACS/selection 1
/ACS/valid_o 1
/ACS/path_cost 11110010
/ACS/path_cost_0 11001101
/ACS/path_cost_1 11110010

Fig.8. ACSU

ACSU at the state 0 (01) Path_0 11001100 at the state 1(10) Path_1 11001101.

TRACE BACK UNIT

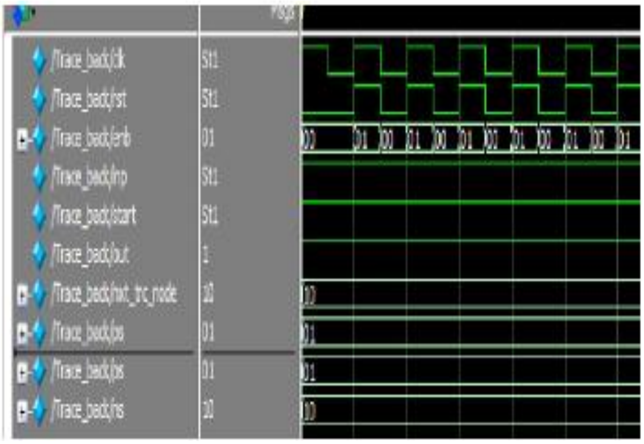

/Trace_back/clk St1
/Trace_back/rst St1
/Trace_back/enb 01
/Trace_back/inp St1
/Trace_back/start St1
/Trace_back/out 1
/Trace_back/nxt_trc_node 10
/Trace_back/ps 01
/Trace_back/ps 01
/Trace_back/ns 10

Fig.9. TBU

TBU Clk-1, Rst-0, Enable-00, Present State-01, Next State-10.

VITERBI TX_RX TOP MODULE

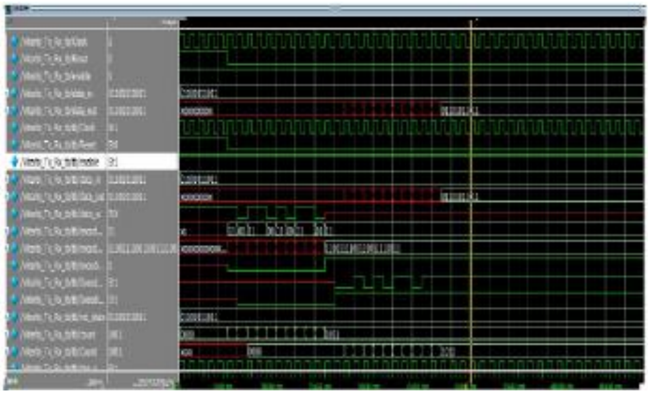


Fig.10. VITERBI TX_RX

Simulation results show input 0110101011011 is encoded using Finite State machines. It generate the data to be transmitted through the channel-result is binary data bits convolutionally encode the data-result is channel symbols. After all of the inputs have been presented to the encoder, the output sequence 11 00 11 11 00 11 00 11 11 00 1 1

Viterbi decoder receives a bit stream containing information about the reliability of each received symbol. Trace-Back unit restores an (almost) maximum-likelihood path from the decisions made by PMU. Since it does it in inverse direction, a viterbi decoder comprises a FILO (first-in-last-out) buffer to reconstruct a correct order.

8. CONCLUSION

Viterbi Algorithm allows safe data transmission via error correction and its original message can be recovered accurately without any noise. It was concluded from that if trace back is started after going deeper into trellis diagram then more accurate data can be achieved but it results in complex hardware design and latency in the received signal. Viterbi algorithm of any rate can be designed using same basic principles and its techniques.

REFERENCES

[1] Jinjin He, Huaping Liu, “High-speed Low-power Viterbi Decoder design for TCM Decoders”, *IEEE Trans. VLSI*, Vol. 20, Apr 2012.

[2] Anubhui Khare, Manish Saxena, Jagdish Patel, “FPGA Based Efficient Implementation of Viterbi Decoder”, *International Journal of Engineering and Advanced Technology (IJEAT),* Volume-1, Issue-1, October 2011.

[3] J. He, Z.Wang, and H.Liu, “An efficient 4-D 8PSK TCM decoder architecture,” *IEEE Trans. Very Large Scale Integer. (VLSI) Syst.*, vol.18, no.5, May 2010.

[4] J. He, H. Liu, and Z. Wang, “A fast ACSU architecture for viterbi decoder using T-algorithm,” *in Proc. 43rd IEEE Asilomar Conf. Signals, Syst. Comput.,* Nov. 2009.