Implementation Analysis of adaptive Viterbi Decoder for High Speed Applications

P. Subhashini
M. Tech Student
Kakinada Institute of Engineering & Technology, JNTUK

D. R. Mahesh Varma
Assistant Professor, ECE
Kakinada Institute of Engineering & Technology, JNTUK

Y. David Solomon Raju
Associate Professor, ECE
Holy Mary Institute of Technology & Science, JNTUH

ABSTRACT
The demand for high speed, low power and low cost for Viterbi decoding especially in wireless communication are always required. Thus this paper presents the design of an adaptive Viterbi decoder that uses survivor path with parameters for wireless communication in an attempt to reduce the power and cost and at the same time increase the speed. Viterbi Algorithm is the optimum-decoding algorithm for convolutional codes and has often been served as a standard technique in digital communication systems for maximum likelihood sequence estimation. The Add-Compare-Select (ACS) and Trace Back (TB) units and its sub circuits of the decoder have been operated in deep pipelined manner to achieve high transmission rate. In this paper register exchange based survivor unit is used as they have better throughput when compared to trace back using memory. Branch metric is calculated for either upper or lower half of trellis, which leads to reduction of power consumption. The Trellis code structure is divided into two segments. The first segment of the Trellis works in normal Viterbi mode while the second works in modified T-algorithm. The designed Adaptive Viterbi Decoder is able to detect and correct up to four errors. The design is optimized with respect to time, area and power and the netlist is generated. The netlist obtained after synthesis undergoes the physical design process. The synthesized circuits are placed and routed in the standard cell design environment and implemented on a Xilinx FPGA device.

Keywords- Convolutional Encoder, Adaptive Viterbi Decoder, Survivor Path, FPGA Implementation

1. INTRODUCTION
Convolution codes are the most widely implemented type of forward error control (FEC) coding in wireless communication system. The modest complexity and good performance have made the Viterbi algorithm the preferred decoding method for convolutional codes to overcome transmission errors. Convolutional encoder with Viterbi decoder acts as powerful method for Forward Error Correction (FEC). Viterbi Algorithm is an optimum decoding Algorithm for convolution code if is applied to transmission in additive white Gaussian noise channel. The Viterbi algorithm (VA) occupies large memory, computational resources, and power consumption to address this issue adaptive Viterbi algorithm (AVA) is introduced.

This paper deals with the implementation analysis of Adaptive Viterbi Decoder for high speed applications. The aim of the paper is to design an Adaptive Viterbi Decoder suitable for High Speed Applications like WLAN, Wi-Max, 3G, etc. by using the Viterbi algorithm as an error correcting code in the decoding of convolution codes.

1.1 Methodology
The literature review on Adaptive Viterbi Decoder is to be carried out by referring journals, books, websites and related documents. Design specifications of Adaptive Viterbi Decoder will be formulated based on application and reviewed literature. All the sub-modules of the Adaptive Viterbi Decoder block to be identified. All the sub-modules of the Adaptive Viterbi Decoder block are to be modeled in Verilog HDL and simulated using ModelSim. A test bench for verifying the complete Adaptive Viterbi Decoder block is to be developed in Verilog HDL using ModelSim. Design synthesis to be performed using Xilinx ISE.

1.2 Significance
The probability of error can be reduced by transmitting more bits than needed to represent the information being sent, and convolving each bit with neighboring bits so that if one transmitted bit got corrupted, enough information is carried by the neighboring bits to estimate what the corrupted bit was. This approach of transforming a number of information bits into a larger number of transmitted bits is called channel coding, and the particular approach of convolving the bits to distribute the information is referred to as convolution coding. Convolutional codes are frequently used to correct errors in noisy channels. They have rather good correcting capability and perform well even on very bad channels (with error probabilities of about 10^{-3}). Convolutional codes are extensively used in satellite communications.

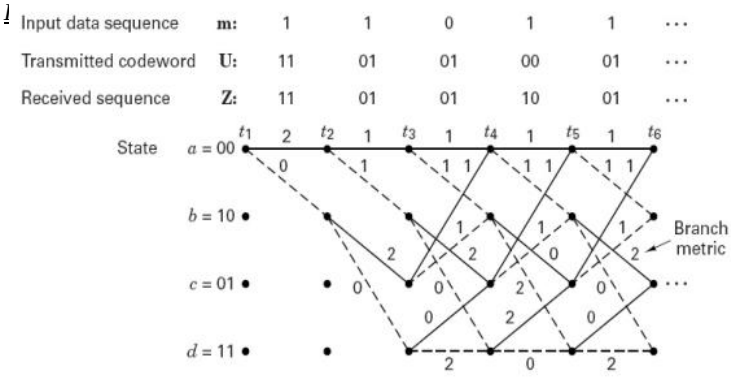
Although convolutional encoding is a simple procedure, decoding of a convolutional code is much more complex task. Viterbi decoding is an optimal (in a maximum-likelihood sense) algorithm for decoding of a convolutional code. Its main drawback is that the decoding complexity grows exponentially with the code length. To overcome come this we use Adaptive Viterbi Decoder. Using convolutional codes together with Adaptive Viterbi algorithm can provide a large coding gain and good performance (a fully parallel Viterbi decoder implemented in hardware is able to process one input pair of bits per clock).

The best way of decoding against random errors is to compute the received sequence with every possible code sequence. This is called maximum likelihood (ML) decoding. The criterion for deciding between two paths is to select the one having the smaller metric. The rule maximizes the probability of a correct decision. The Viterbi algorithm operates on a state machine assumption. That is, at any time the system being modelled is in some state. There are a finite number of states, however large, that can be listed. Each state is represented as a node. Multiple sequences of states (paths) can lead to a given state, but one is the most likely path to that state, called the "survivor path". This is a fundamental assumption of the algorithm because the algorithm will examine all possible paths leading to a state and only keep the one most likely. This way the algorithm does not have to keep track of all possible paths, only one per state. A second key assumption is that a transition from a previous state to a new state is marked by an incremental metric, usually a number. This transition is computed from the event. The third key assumption is that the events are cumulative over a path in some sense, usually additive. So the crux of the algorithm is to keep a number for each state. When an event occurs, the algorithm examines moving forward to a new set of states by combining the metric of a possible previous state with the incremental metric of the transition due to the event and chooses the best. The incremental metric associated with an event depends on the transition possibility from the old state to the new state. It has widely deployed in many wireless communication systems to improve the limited capacity of communication channel. The complexity of Viterbi Algorithm (VA) is proportional to number of states in the decoding trellis, where the number of state 2^k and k is total number of encoder memory bit used in the encoder for the convolution code. If k larger, will leads to gain high correction capability and large circuit, power and low speed decoding.

2. VITERBI DECODER

The receiver can deliver either hard or soft symbols to the Viterbi decoder. A hard symbol is equivalent to a binary +/-1. A soft symbol, on the other hand, is multileveled to represent the confidence in the bit being positive or negative. For instance, if the channel is non-fading and Gaussian, the output of the matched filter quantified to a given number of bits is a suitable soft input. In both cases, 0 is used to represent a punctured bit. In case of hard decision demodulation, data is demodulated into either 1s or 0s, or quantized into two levels only. The process described above makes a hard binary decision about each incoming bit and then uses only the Hamming distances. This simplifiers the hardware, but does not result in optimal performance.

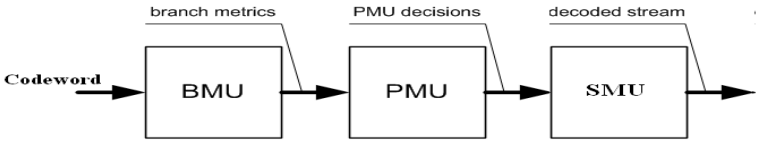
Figure 2.1 Block diagram of Viterbi decoder [4]



The Working of Viterbi decoder in term of block diagram and trellis diagram is show in Figure 2.1 and 2.2.

Figure 2.2 Trellis diagram of Viterbi decoder

For hard decision decoding, the Viterbi algorithm uses the



hamming distance to find the branch metric and path metric. Codeword is given to branch metric unit. Branch metric unit's function is to calculate branch metrics, which are Hamming distances between every possible symbol in the codeword and the received symbol. Path metric unit summarizes branch metrics to get metrics for 2K − 1 path, one of which can eventually be chosen as optimal. Survivor memory unit can be trace-back process or register exchange method, where the survivor path and the output data are identified. The error probabilities achieved by Viterbi algorithm depends on the code, the rate of the code, its free distance, channel SNR and demodulation Quantized output. The quality of Viterbi decoder design is mainly measured by three criteria

- Coding gain
- Throughput
- Power dissipation

2.1 Adaptive Viterbi Algorithm

The VA examines all possible paths in trellis graph and determines the most likely one. The AVA only keeps a number of the most likely state instead of the whole of 2k-1 state, where is constraint length of convolution encoder. The rest of the other states are all discarded. The selection is based on the likelihood or metric value of the path, which for hard decision is the hamming distance and a soft decision decoder is Euclidean distance.

- Every Surviving path at trellis level L -1 is extended and its successors at level l are kept if their path metric is smaller or equal to d_m+T, where dm is the minimum path metric of the surviving path at stage L-1, T is discarding threshold configure by the designer.
- The total number of survivor path per trellis stage is up bounded to fixed number, which is pre-set prior to the start of the communication. Generally this will be same as number of states in decoder.

2.2 Adaptive Viterbi Decoder (AVD)

Figure 2.3 shows the data flow diagram of an adaptive Viterbi algorithm, which adds two functional

blocks, including the best winner search and non survivor purge, into the original Viterbi algorithm.

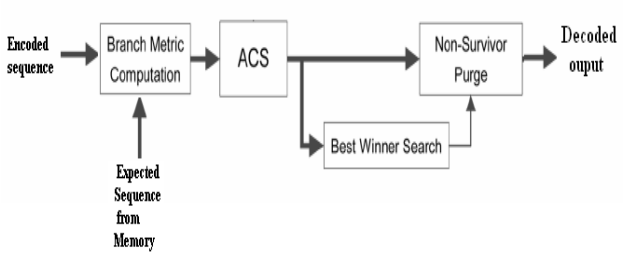


Figure 2.3 Block diagram of Adaptive Viterbi decoder [6]

Codeword is applied to branch metric computation unit. It calculates branch metric by comparing with expected symbol. ACS updates path metric by cumulative accumulation of branch metric. Best winner search determines final winner and give it non survivor purge unit. It deletes all paths expect winner. The first unit is called branch metric unit BMU is the simplest block in the Viterbi decoder design. 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.

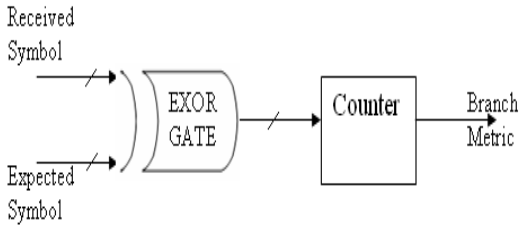


Figure 2.4 Block diagram of Branch Metric Unit **[7]**

Block diagram of BMU (Branch metric unit) is shown in Figure 2.4. The BMU calculates the branch metrics from the input data. For hard decision BMU calculate everything in term of hamming distance. Hamming distance between the received Codeword and the expected is calculated by compares the received code symbol with the expected code symbol and counting the number of different bits. BMC (branch metric computation) unit is to calculate the branch metrics which are then moved to the ACS (add compare select) unit. The major task of the ACS is to calculate the metrics and selected paths. The add-compare-select (ACS) unit recursively accumulates the branch metrics to path metrics for all the incoming paths of each state and selects the path with minimum path metric as the survivor path. An ACS module is shown in Figure2.5. The two adders compute the partial path metric of each branch, the comparator compares the two partial metrics, and the selector selects an appropriate branch. ACS units determine their own local winners, the best winner search block finds the one having the best (minimum) path metric among all the winners, and the non survivor purge block deletes the local winners.

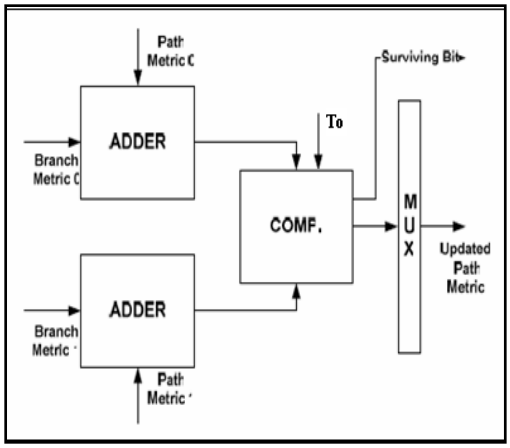


Figure 2.5 Add-Compare-Select Unit [7]

Survivor Unit of adaptive Viterbi decoder based on either an RE (register exchange) or TB (trace-back) design style. The TB method takes up less area but requires much more time as compared to RE method because it needs to search or trace the survivor path back sequentially. The trace back approach is generally a lower power alternative to the register exchange method.

Block diagram of Trace back unit is shown in Figure 2.6 .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 Path Metric (Global winner).

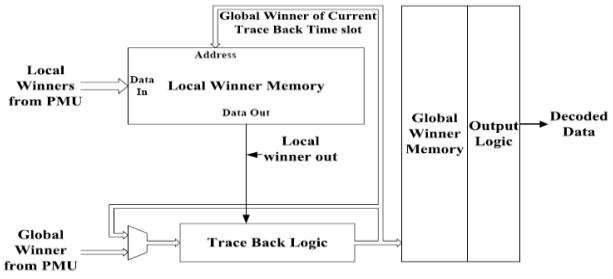


Figure 2.6 Block diagram of Trace back unit [7]

The Register Exchange (RE) method is the simplest conceptually and a commonly used technique. In this technique, the Trellis Window is constructed of a bank of registers connected in the same manner as the trellis diagram. The data path in Register Exchange is shown in

Figure 2.7. 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. In this method a register assigned to each state contains information bits for the survivor path from the initial state to the current state. Bold arrow indicate Global winner path. This method eliminates the need to trace back since the register of the final state contains the decoded output sequence. However, this method results in complex hardware due to the need to copy the contents of all the registers in a stage to the next stage.

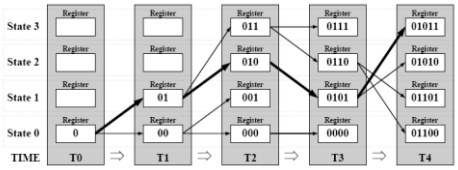


Figure 2.7 Block diagram of Register Exchange method [7]

Trellis codes are characterized by a parameter called the free distance (Dfree). Dfree is defined as the minimum Hamming distance between any two distinct code sequences. In other words, it is the minimum sum of Hamming distances between the codeword mapped on the branches of two paths in the Trellis, which first diverge and then merge back. Truncation length (**τ0**) is given by expression τo= T1 + t .Where T1 is best (smallest) path metric of all the states and t is a fixed value given by expression [dfree-1]/2]. Truncation length (**τ0**) controls the average number of path stored per trellis at each stage and the frequency with which trace back length through the trellis is performed. The reason energy consumption can be significantly reduced for adaptive by reducing truncation length is mainly because these parameters greatly affect the number of times path memory is accessed. Actually, the number of time nearly all calculations are performed per trellis stage is proportional to average number of surviving path per stage, which is controlled by truncation length.

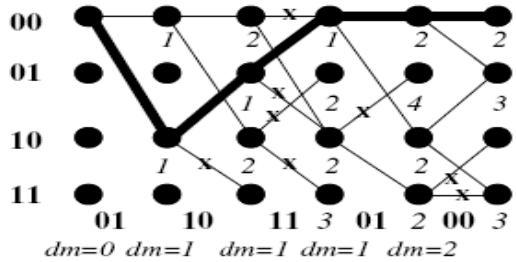


Figure 2.8 Trellis Graph of Adaptive Viterbi decoder [8]

Figure2.8 shows Trellis graph of Adaptive Viterbi decoder. The symbol X represent discarded paths and bold line represent the decision path by the Adaptive Viterbi algorithm. It can been seen that at each stage of trellis , the number of survivor state is smaller than the Viterbi Algorithm (2^{k-1}) and get the same decision path as the Viterbi Algorithm.

To implement a convolutional coding system, the appropriate code parameters, R and K, should be decided first. In addition, the design decisions for the decoding system, such as the quantization levels etc. Decoding schemes with hard decision is a simplified version of a modulator, AWGN Channel and demodulator. Figure 2.9 describes binary memory less channel where the input and output of transmission channel are bi-level. In the Figure 2.8, p (1/0) is probability that a 0 is transmitted and a 1 is received and in general it depends on the signal to noise ratio in the channel and the modulation type.

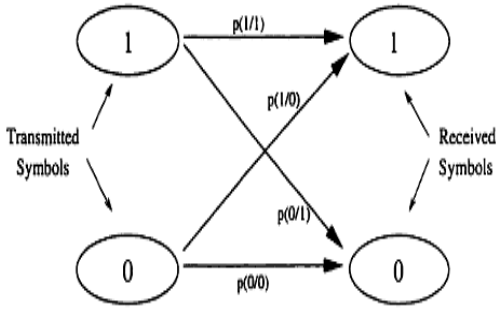


Figure 2.9 Binary memory less channel [8]

2.3. Low-power design techniques for Viterbi Decoder

Low-power design can be performed at the architecture level, the gate level and the switch level. At the architecture level, different architectures, such as parallel and pipelined architecture or transformations may be considered for low power dissipation. The reduction of switching activity at nodes of a circuit, which directly affects the power dissipation, is the major focus at the gate level. Parameter adjustment and transistor sizing can be applied at the switch level. For a standard cell approach, there is no control over internal circuitry (i.e. transistors) of a cell. Therefore low power design can be considered only at the architecture and the gate levels. In this thesis, we consider low power design techniques mainly at the gate level. A major reduction in power dissipation can be achieved by reducing the switching activity on which a designer has more control. A careful description of the circuit in a high-level hardware description language can yield a circuit with a lower switching activity. At this level, one of the most powerful power reduction techniques called the clock gating can be applied. Some blocks of a circuit are used only during a certain period of time. The clock input to the blocks can be disabled to eliminate unnecessary switching when the blocks are not in use. By applying clock gating, the dynamic power dissipation in a design can be reduced to a great extent.

2.4 Other Decoding Algorithms

There exist four basic convolutional codes decoding techniques: sequential, threshold, maximum-likelihood and the Viterbi algorithm. The sequential algorithm can provide very strong correcting capabilities while it needs relatively large memory, which strongly depends on communication channel error density. The threshold algorithm is extensively good for channels with mid to good signal to noise ratios (SNR).mPrior to the discovery of the VA, a number of other algorithms had been proposed for decoding convolutional codes. These were sequential decoding algorithm proposed by Wozen

craft and subsequently modified by Fano. The Fano sequential algorithm searches for most probable path through the trellis by examining one path at a time. In this algorithm an additional negative constant is added to the each branch metric. The value of this constant is selected such that the metric of the correct path will increase on the average, while the metric of incorrect path will decrease on the average. By comparing the metric of a candidate path with an increasing threshold, Fano's algorithm detects and discards incorrect path. Its error performance is comparable to that of Viterbi decoding. In comparison with Viterbi decoding algorithm, sequential decoding has larger decoding delay. On the positive side, sequential decoding requires less storage than Viterbi decoding and hence it is more attractive for convolutional codes with large constraint length.

Another type of sequential decoding algorithm is called a stack algorithm. In contrast to VA, this keeps track of $2^{(k*(K-1)}$ paths and corresponding metrics, the stack sequential decoding algorithm deals with fewer paths and their corresponding metrics. In a stack algorithm, more probable paths are ordered according to their metrics, with the path at the top of the stack having largest metric. At each step of the algorithm, only the path at the top of the stack is extended by one branch. In comparison to VA, the stack algorithm requires fewer metric computations, but this computational saving is offset to a large extent by the computations involved in reordering the stack after taking into account of every iteration.

RE-Based Design: Assuming each trellis state only has two incoming/outgoing branches. RE-based design solution by introducing clock-gating into the RE array followed by majority vote unit. The validity bits from the modified ACS units array directly control the clock-gating for power reduction. Since the average number of survivors can be much less than the total number of trellis states in adaptive Viterbi decoding, this may lead to significant power savings.

TB-Based Design: To support high throughput in a state-parallel decoder, the TB-based survivor memory unit must provide large enough memory access bandwidth. One natural approach to increase the access bandwidth is to use a bank of memories that can be accessed concurrently.

State-parallel relaxed adaptive Viterbi decoder architectures that can readily transform the reduced computational complexity at the algorithm level to reduced switching activities and hence power consumption at the hardware level.

Specification of Adaptive Viterbi Decoder

- Constraint length K =3
- Code Rate r = ½
- Generator polynomials G0=171_8, G1=133_8
- No of state = 4
- Free distance dfree =10
- Application target = 802.11 a receiver
- Survivor unit type = Register Exchange

3. DESIGN

According to Viterbi there are 3 assumptions. First any system can be modelled in some state. There are a finite number of states, however large, that can be listed. Each state is represented as a node. Second a transition from a previous state to a new state is marked by an incremental metric. Third moving forward to a new set of states by combining the metric of a possible previous state with the incremental metric of the transition due to the event and chooses the best. The incremental metric associated with an event depends on the transition possibility from the old state to the new state. First Transition table is prepared from convolution encoder for the specification and stored in memory. This transition table describes about next possible outcome and there encoded value for that particular transition for given input state. Since our encoding rate is ½ every state while have two possible outcomes and similar every state while have two incoming branch. It's works like a controller for whole trills. Transition table k=3 at rate ½ displayed for example in table 3.1.

Table 3.1 Transition table k=3 at rate ½

STATE	NEXT POSSIBLE OUTCOME 1	ENCODED VALUE FOR OUTCOME 1	NEXT POSSIBLE OUTCOME 2	ENCODED VALUE FOR OUTCOME 2
00	00	00	10	10
01	00	11	10	01
10	01	11	11	01
11	01	00	11	10

Branch metrics unite used to compare encoded inputs applied to Viterbi decoder and encoded values for transition from one stage of trills to next stage based on this branch weight is calculated. It feed as input Add compare and select unit. Add Compare and select (ACS) works on every state of stage on trills. Two incoming branch of state is decided based on transition table. ACS unit while update path metric for incoming branch and eliminate path with higher path metric or if path metric value greater than $\boldsymbol{\tau 0}$ value. ACS and BMU are clubbed together to form a butterfly module. So for 64 stages there will be 64 butterfly modules. Best winner search unit works on stage of trills. Work of ACS unite is to find local winner on every node (state) of trills. Work of Best winner search unit is find final winner at end of operation.

Finally trills is formed by integrate all above module together. In trills at first stage path metric set to non-zero value for all states expect stage 0(path metric is set to zero). Similar if ACS unit want to delete path then path metric of that state is set to maximum value. By doing this path get automatic eliminated at the next stage of trills. In order to keep sufficient error correction capability, the length of the path (trace back length) should be $5(K-1)$.In this design value of k is 7 so trace back length should be thirty. Decoded can be done either by Trace Back method (using memory) or by Register Exchange method. Each method have there own advantages and disadvantage. Trace Back unit start working once best winner give as final winner. The current best state is used to predict the previous state by referencing the corresponding value of the first column of the Trellis Window. This process is repeated for each computed state and the corresponding column till the end of the table. So it doubles the amount of time taken for decoded data.

The RE technique is a straightforward technique for managing the decision vectors. In this technique, the Trellis Window is constructed of a bank of registers connected in the same manner as the trellis diagram. The newest decision hits are inserted in the left column of the Trellis Window as the oldest bits out at the right of the window. Decoding is done at end of trills once best winner search chosen final winner. Its increases number of registers in design result increase in area and power. In a RE architecture the trellis window is the main contributor for power consumption. This is because the data is being shifted through the complete table which results in a huge transition activity. Size of register in Register Exchange based decoder depends on two factors one stage of Trills and second Length of trace back.

Figure 3.1 R.E based Viterbi architecture [12]

Adaptive Viterbi decoder is functionally same as Viterbi decoder when no error is introduced. Once error is introduced in input Adaptive part come into picture. It check every node for path metric value higher than τo .If path is found its eliminated else usual action take place on node like normal Viterbi decoder. So data travel in trills of Adaptive Viterbi decoder entire different its counterpart Viterbi decoder in presents of errors. Where τo= T1 + t [T1 is best (smallest) path metric of all the states and t is a fixed value given by expression [dfree-1]/2]. So Value of τo= 0 + 5= 5. In design τo value should be chosen greater or equal to five.

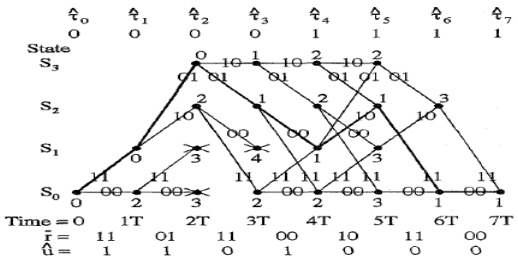
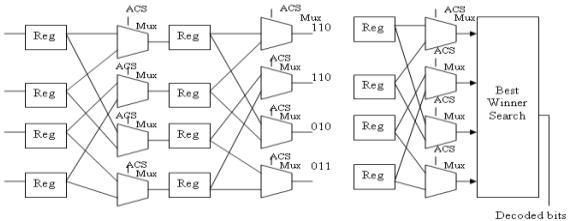


Figure 3.2 Trellis diagram of AVD for k=3 at rate ½ with τ0 =3

Any path passing a state with metric greater than τo can be a best path. This implies that the number of errors appearing in the channel exceeds the error-correcting capability for the convolutional code. Hence, the exclusion of paths which pass states with metrics greater than τ_o will barely affect the error probability of decoding. Trellis diagram of AVD for k=3[7, 5] at rate ½ with τ0 =3 shown in Figure 3.2.Trilles Structure for AVD is based on Modified T-algorithm. In Trills path with path greater or equal to 3 are eliminated. Final to reduce the number of ports in system serial in parallel out and parallel in serial out been added to start and end of design.

4. RESULTS

A binary sequence is to applied convolution encoder. Sequence generated as result of convolution encoder is applied as validation input to Adaptive Viterbi decoder hardware model. Output of hardware model is compare back with input of convolution encoder. To check error correcting capability of Adaptive Viterbi decoder some error are introduced in various part of sequence which is applied as input of Adaptive Viterbi decoder and obtained output once again compared back with input of convolution encoder



4.1 Encoder output

The encoder module takes one bit as input and it encodes it into two bit code. We can see simulation result of encoder block in the below figure.

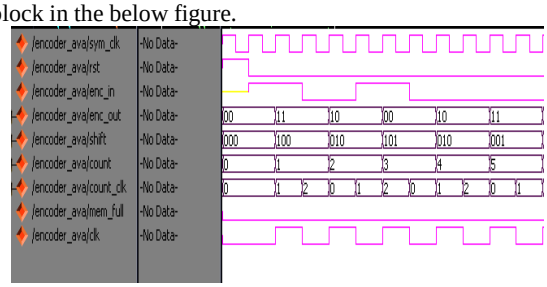


Figure 4.1 Simulation Results for Encoder Block

We can observe from the above waveform how the encoder takes one bit input (enc_in), and encodes it into two bit code (enc_out).

4.2 Decoder Output

The decoder takes two bit encoded data as input and decodes it into one bit data. We can see the simulation result of the decoder block in the below figure.

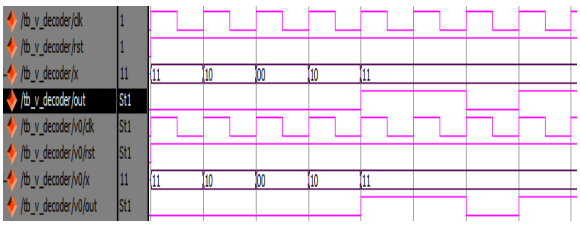


Figure 4.2 Simulation result for the decoder block

We can observe from the above waveform how the decoder takes two bit input(x) and decodes it into one bit (out) data.

4.3 Synthesis Results

The developed Adaptive Viterbi Algorithm is simulated and verified their functionality. Once the functional verification is done, the RTL model is taken to the synthesis process using the Xilinx ISE tool. In synthesis process, the RTL model will be converted to the gate level netlist mapped to a specific technology library. This Adaptive Viterbi Algorithm design is implemented on FPGA (Field Programmable Gate Array) family of Spartan 3E. Here in this Spartan 3E family, many different devices were available in the Xilinx ISE tool. In order to implement this Adaptive Viterbi Algorithm the device named as "XC3S500E" has been chosen and the package as "FG320" with the device speed as "-4". The design of Adaptive Viterbi Decoder is synthesized and its results are analyzed as follows.

4.4 RTL Schematic

The RTL (Register Transfer Logic) can be viewed as black box after synthesize of design is made. It

shows the inputs and outputs of the system. By double-clicking on the diagram we can see gates, flip-flops and MUX. The below figure 4.3 shows the top level block diagram that contains the primary inputs and outputs of the design.

Figure 4.3 RTL Top Level Block

4.5 Device Utilization Summary

This device utilization includes the following.

- Logic Utilization
- Logic Distribution
- Total Gate count for the Design

nm Project Status (02/17/2010 - 13:48:50)

Project File:	nm.ise	**Current State:**	Programming File Generated
Module Name:	v_decoder	**• Errors:**	No Errors
Target Device:	xc3s500e-4fg320	**• Warnings:**	14 Warnings
Product Version:	ISE 10.1 - Foundation Simulator	**• Routing Results:**	All Signals Completely Routed
Design Goal:	Balanced	**• Timing Constraints:**	All Constraints Met
Design Strategy:	Xilinx Default (unlocked)	**• Final Timing Score:**	0 (Timing Report)

nm Partition Summary

No partition information was found.

Device Utilization Summary

Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	24	9,312	1%	
Number of 4 input LUTs	87	9,312	1%	
Logic Distribution				
Number of occupied Slices	48	4,656	1%	
Number of Slices containing only related logic	48	48	100%	
Number of Slices containing unrelated logic	0	48	0%	
Total Number of 4 input LUTs	87	9,312	1%	
Number of bonded IOBs	5	232	2%	
Number of BUFGMUXs	1	24	4%	

Detailed Reports

Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	Wed Feb 17 13:44:06 2010	0	14 Warnings	17 Infos
Translation Report	Current	Wed Feb 17 13:47:43 2010	0	0	0
Map Report	Current	Wed Feb 17 13:47:55 2010	0	0	2 Infos
Place and Route Report	Current	Wed Feb 17 13:48:19 2010	0	0	2 Infos
Static Timing Report	Current	Wed Feb 17 13:48:25 2010	0	0	3 Infos
Bitgen Report	Current	Wed Feb 17 13:48:47 2010	0	0	0

Figure 4.4 Device utilization summary

The device utilization summery is shown above in which its gives the details of number of devices used from the available devices and also represented in %. Hence as the result of the synthesis process, the device utilization in the used device and package is shown above.

4.6 Timing Summary

Speed Grade: -4

Minimum period: 6.290ns (Maximum Frequency: 158.983MHz)

Minimum input arrival time before clock: 3.074ns

Maximum output required time after clock: 23.829ns

Maximum combinational path delay: No path found

In timing summery, details regarding time period and frequency is shown are approximate while synthesize. After place and routing is over, we get the exact timing summery. Hence the maximum operating frequency of this synthesized design is given as 18.970 MHz and the minimum period as 52.719 ns. OFFSET IN is the minimum input arrival time before clock and OFFSET OUT is maximum output required time after clock.

The developed Adaptive Viterbi Decoder is modelled and is simulated using the ModelSim tool. The simulation results are discussed for encoder and decoder blocks. The RTL model is implemented using the Xilinx ISE tool in Xilinx Spartan 3E FPGA and the synthesis results are discussed with the help of generated reports.

5. CONCLUSION

In this work, a high speed implementation of an adaptive Viterbi decoder which uses modified T-algorithm is presented .The use of error-correcting codes has proven to be an effective way to overcome data corruption in digital communication channels. Some of the conclusions drawn from the design are as under below. The Register exchange method and the AVA incorporated in design resulted in a throughput of 80Mbps. A high throughput has been achieved by using 32 butterfly modules which operate in parallel. Power and area has been reduced by dividing the Trellis Coding structure into two segments. Significant amount of power has been reduced in the design by modifying branch metric architecture.

Implementation of hard decision adaptive Viterbi decoder for constraint length k=7 at rate 1/2 is discussed here .Same work can be extended for soft decision based adaptive Viterbi decoder which suits multiple quantization level by modifying Branch metric computation unit accordingly.

6. REFERENCES

[1] Bernard Sklar, "*Digital communication Fundamentals and Applications*", 2nd edition, Prentice Hall, ISBN:81-7808-373-6, 2001.

[2] Michael Purser, "*Introduction to Error- correction codes*", Artech House INC, ISBN: 0-89006-784-8, 1996.

[3] Fei Sun and Tong Zhang , "*Low-Power State-Parallel Relaxed Adaptive Viterbi Decoder*" , IEEE Transactions on Circuits and systems , Vol. 54, Page(s)-1060-1069, No. 5, May 2007.

[4] Rex Andrew Antony," *An Adaptive threshold strategy for soft decision Viterbi Decoder*", Dalhouse University, December 2002

[5] QIN Xiang-Ju'.', ZHU Mmg -Cheng', WEI Zhong-Yi2, CHAO Du', "*An Adaptive Viterbi Decoder Based on FPGA Dynamic Reconfiguration Technology*", IEEE International Conference on Field-Programmable Technology 2004, Vol. 10, Page(s)-6-8 December, 2004.

[6] Man Guo, M. Omair Ahmad, M.N.S. Swamy, and Chunyan Wang , "*A Low-Power Systolic Array-Based Adaptive Viterbi Decoder and its FPGA Implementation*", International Symposium on Field-Programmable Technology 2003, Vol 2, Page(s)- 276 - 279, 25-28 May 2003.

[7] Abdulfattah M. Obeid, Alberto Garcia, Mihail Petrov, Manfred Glesner ,"*A Multi –path high speed Viterbi decoder*" , Proceedings of the 2003 10th IEEE International Conference on Electronics, Circuits and Systems, 2003. ICECS 2003, Vol 3, Issue, 14-17 Page(s): 1160 – 1163, December 2003

[8] Samir Palnitkar, "*Verilog HDL – A Guide to Design and Synthesis*", 2nd edition, Prentice Hall, ISBN: 0-13-044911-5 2003.

7. AUTHORS BIOGRAPHY

P. Subhashini received B.E. in Electronics and Communication Engg, from S. R. K. R.E ngg college, Bhimavaram, Andhra University Visakhapatnam. Presently pursuing M. Tech in VLSI Design, Kakinada Institute of Engineering & Technology, Korangi, JNTU Kakinada. Her fields of study were Low Power VLSI, Digital signal processing, as well as programmable logic and digital circuits.

D. R. Mahesh Varma working as Assistant Professor, Kakinada Institute of Engineering & Technology, Korangi, JNTU Kakinada. His fields of study were Communications, VLSI, Digital signal processing, as well as programmable logic and digital circuits.

David Solomon Raju. Y pursuing Ph. D. from Rayalseema University, Kurnool. He was working as Associate Professor in the Department of Electronics and Communication Engineering at Holy Mary Institute of Technology and Science, affiliated to JNTUH from 2007 to till date. Received B. Tech. degree from V. R. Siddhartha Engineering College, affiliated to Nagarjuna University in 2000, M. Tech. Degree from JNTU, Hyderabad in 2007, and His fields of study were VLSI, Digital signal processing, as well as programmable logic and digital circuits. His research interests include Image Processing techniques and wavelet theory applied to communication networks, network security. He has authored several technical publications published in conference proceedings and has been an invited speaker at many national conferences. He has guided many under graduate and post graduate students to present papers at national and international conferences. He is a member of the ISTE.