IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

FPGA Implementation of High Speed Convolutional Encoder and Viterbi Decoder for Software Defined Radio

[1]Vidhya Shree.N , [2]R. Krishna,

[1]PG Student, [2]Assistant Professor

[1]VLSI and Embedded Systems,

[1]Bangalore Institute of Technology, Bengaluru, INDIA

Abstract: The data transmission in any wireless communication system is affected by attenuation, interference, noise and distortion which affect the receiver's ability to receive the correct information. The Convolution encoder and Viterbi decoder are good forward error detection and correcting codes for a channel, which is affected by noise. The convolution encoder is used for correction of errors at the receiver end. Software defined radio adapts various modulation schemes, encoding techniques by changing its configuration. SDR reduces the cost complexity and provides the flexible communication system. This paper presents the implementation of convolution Encoder and Viterbi Decoder with a constrained length of 3 and a code rate of ½ by using Spartan and virtex –families of FPGA. This paper proposes the Viterbi decoder architecture which optimizes the critical path to achieve higher speeds. The design of Viterbi is carried out in MATLAB and verified. RTL coding is done using Verilog HDL, Xilinx Spartan Series FPGA is used for implementation. Modelsim and Vivado is used for simulation functional and timing simulations.

Index Terms **– Viterbi decoder , FPGA , Software Defined Radio.**

I. INTRODUCTION

Viterbi decoder is an important block in the software defined radio transceiver. Convolutional codes are the forward error correcting codes which has been widely used in digital communication systems to make the transmission more reliable. In order to transmit the data over a noisy channel convolutional codes offer better results in comparison to block codes. In convolutional encoding extra bits are added to the input bits to reduce the error probability. One of the efficient methods of decoding the convolutional codes is the viterbi decoder . Branch Metric Unit (BMU), Add Compare Select Unit (ACSU) and Trace Back Unit (TBU) are the three main units in viterbi decoder. Hamming distance between the received symbol and the possible output symbol is computed by using the branch metric unit. Add Compare Select unit is used to find the minimum distance from one of the four states. Trace back unit traces back the minimum path to find the original bits that was given to the encoder at the transmitter side. Software defined radio (SDR) is a radio communication system where components that have been traditionally implemented in hardware (e.g. mixers, filters, amplifiers, modulators/demodulators, detectors, etc.) are instead implemented by means of software on a personal computer or embedded system.[1] While the concept of SDR is not new, the rapidly evolving capabilities of digital electronics render practical many processes which were once only theoretically possible. A basic SDR system may consist of a personal computer equipped with a sound card, or other analog-to-digital converter, preceded by some form of RF front end. Significant amounts of signal processing are handed over to the general-purpose processor, rather than being done in special-purpose hardware (electronic circuits). Such a design produces a radio which can receive and transmit widely different radio protocols based solely on the software used.

II. LITERATURE SURVEY

- In 2015,*Amruta J. Mandwale ; Altaf O. Mulani* proposed the impacts of the information sent over any correspondence channel is influenced because of commotion. Along these lines, for identifying and remedying the mistakes because of channel clamor, encoding and deciphering ought to be performed at the transmitter and recipient end separately.
- In 2017 ,*Swathi1, S. Rajaram* proposed about Software defined radio and it versatile nature to different balance plans, encoding methods by changing its configuration. It has earned incredible consideration in the ongoing years for its flexibility to adjust different procedures without the need of substitution of the current equipment.
- In 1967 ,*Berlekamp* showed effective disentangling calculation for both non-double BCH and Reed-Solomon codes. Berlekamp's calculation takes into account the proficient translating of many blunders all at once utilizing extremely amazing Reed-Solomon codes.
- In 2017,*Ashraf Mahran ,Ramy Samy* proposed an effective adjustment with one of the best unpredictability decrease procedures of the Viterbi interpreting calculation; the versatile Viterbi calculation. In spite of the fact that the utilization of the versatile calculation made it for all intents and purposes obvious to utilize convolutional codes with enormous imperative lengths.
- In 2007,*Sugiyama, Kasahara, Hirasawa and Namekawa* indicated that Euclid's calculation can also be utilized to productively translate BCH and Reed-Solomon codes. Euclid's calculation is a methods for finding the best basic divisor of a couple of numbers.
- In *1999,Bupesh Pandita&et .al* implemented FPGA based 2- way add compare select unit which used constraint length 8, code rate 1/2, Viterbi decoder with generator polynomial as (561,743)g. Special emphasis was given over clock synchronization. They developed an area efficient structure with good throughput of more than 19.2kbps by using VHDL.
- *Yan Sun,Zhizhong Ding* talks about a changed FPGA conspire for the Convolutional Encoder and Viterbi decoder dependent on the 802.11 a guidelines of WLAN is introduced in OFDM baseband handling frameworks.
- In 2012, *Li Li-fu ,Li Hai-wen, Li Hong-liang , Gu Yong-jun* proposed the Viterbi calculations of the tail gnawing convolutional code in TD-LTE framework is presented. Focusing on the translating center for the convolutional codes of TD-LTE framework.

III. EXISTING ARCHITECTURE

The Existing ACS module for decoder (Viterbi) block outline is appeared in Fig.3 below.The encoder generates a pair of bits for each input bit received, this is because the rate is ½, the trace back length for this architecture is 8, and hence 8 pair of bits are generated by the encoder. This 8 pair of bits i.e, 16 bits in total act as the input to Viterbi decoder, which runs a state machine using a counter, at every state it collects 2 bits and performs various tasks such as, finding the error metric, the next state computation, once after the 3rd state there are multiple paths converging at the same state, but the decoder circuit should only pick one among them, this is decided by comparison of cumulative error metric right from the origin. This error matrix is stored in a buffer. Hence the architecture works based on the xor-ing bits for finding the hamming distance i.e. the error metric and using an adder for cumulative addition. Then comes the comparator circuit which acts as the input selection to mux, which decided which path to choose.

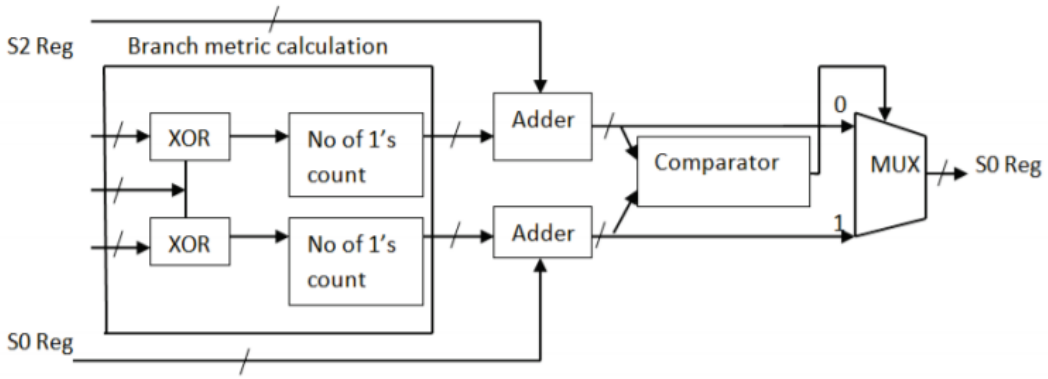

Fig 3 Existing Architecture for ACSU Module

IV. PROPOSED METHDOLOGY

The proposed methodology is shown in Fig.4 and functions of each block is explained ,

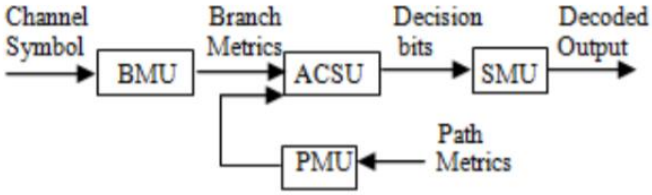

Fig.4 functional diagram of a Viterbi decoder

BMU: Branch Metric Unit
ACSU: Add Compare Select Unit
SMU: survivor memory unit
PMU: Path metric Unit

The Branch Metric Unit

The Branch Metric unit registers the hamming separation between the got input succession and the normal information arrangement. In this paper Branch metric generator depends on a fact table containing the different piece measurements. The PC looks into the n-digit measurements related with each branch and totals them to acquire the branch metric. These outcomes are passed to the way metric update and capacity unit.

ACSU Module

This module is the core part of the Viterbi decoder . ACSU is utilized consistently in the decoder with the end goal of the count of the new branch measurements and furthermore for refreshing the memory components with least branch measurements. The number of times the ACSU circuit utilized is relying upon the limitation length. As the limitation length builds the lattice stages increments subsequently the number of times the ACSU utilized additionally increments.
In our proposed paper same sort of four ACSU are there to figure the state metric for each state at same time.
Also with asynchronous approach one can increase the speed of the system. Hence this approach is used to build the adder that is used in this paper. Other term for asynchronous logic is self-timed.

Survivor Path Storage and Management Module

This module is used for storing and managing survivor path values of the ACS modules. Three RAM blocks work by turns. The following is a detailed description.

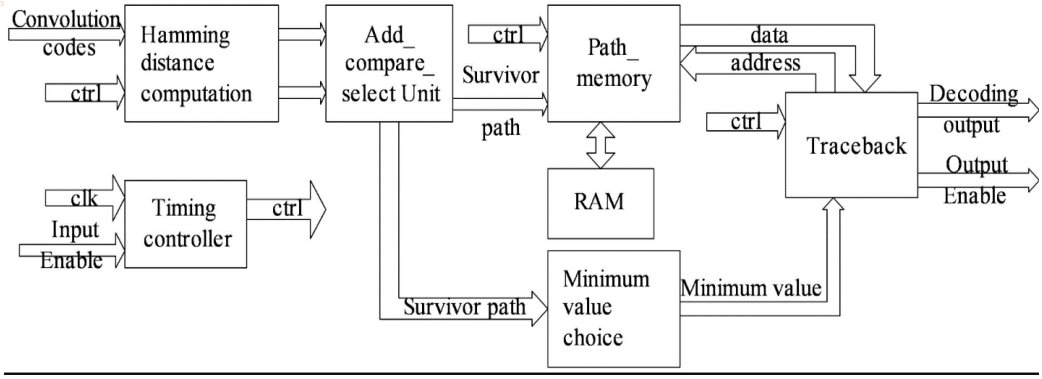

Fig.5 Detailed description of survivor path storage and management module

Step 1, the first RAM block is written to survivor paths for sixty six clock periods which equals to decoding depth, and then it begins to trace-back and decode.
Step 2, the second RAM block begins to be written to survivor paths like the first one. After another decoding depth the second one begins to trace-back and the third one begins to be written to survivor paths.
Step 3, when the third RAM block is full, it begins to trace-back. At this time the first RAM block is free, therefore, the survivor paths can be written to it again.
In this study, decoding and trace-back is performed at the same time. Three RAM blocks are used for this module because the write operation and read operation are not the same start and end. When the decoding depth is reached in the first RAM block, sixty four cumulative hamming distances of all the states are inputted to the minimum value choice module; this module needs thirty three

clock periods to finish the work. In this period the second one is being written. The first RAM block just starts decoding when the second one is full. If the first one that should be written is not free when the write operation of the second one is over, the third RAM block should be provided for the new write operation. Therefore, two RAM blocks cannot be used by turns.

The Path Metric Unit

The way metric is the aggregate of the branch metric of the current information and the way metric of the past emphasis. This takes the branch measurements processed by the BMU and registers the halfway way measurements at every hub in the lattice. These Metrics are put away in the memory components for utilizing them in the following cycles and furthermore to locate the base way metric at the last stage (iteration) of the lattice. The enduring way at every hub is distinguished, and the data succession refreshing and capacity unit.

Trace-Back Module

There are two algorithms of survivor paths storage and management for decoding. One is register-exchange algorithm; the other is trace-back. The principles and hardware implementation of the register-exchange algorithm are more easily. Compared with trace-back, this algorithm has one obvious advantage that is the smaller output delay. But it brings two mortal flaws, one is lots of power consumption when the memory contents of each state are always read and written for updating survivor paths with every clock pulse, the other is too complex internal connection relationships to restrict the FPGA design. So it is rarely used in practical applications. Trace-back is a classic algorithm and widely used, and uses memories to store survivor paths of each state. The former trace-back point is determined by the best initial state and the corresponding survivor paths every clock. The trace-back is completed after thirty six clock periods. The hardware structure of the algorithm is simple to manage and control for FPGA.

Viterbi Algorithm

Viterbi algorithm is called optimum algorithm since it minimizes the probability of error. The Viterbi algorithm can be explained briefly with the following three steps .

1. Weigh the trellis; that is, calculate the branch metrics.

2. Recursively compute the shortest paths to time n, in terms of the shortest paths to time n-1. In this step, decisions are used to recursively update the survivor path of the signal. This is known as add-compare-select (ACSU) recursion.

3. Recursively find the shortest path leading to each trellis state using the decisions from Step 2. The shortest path is called the survivor path for that state and the process is referred to as survivor path decode. Finally, if all survivor paths are traced back in time, they merge into a unique path, which is the most likely signal path.

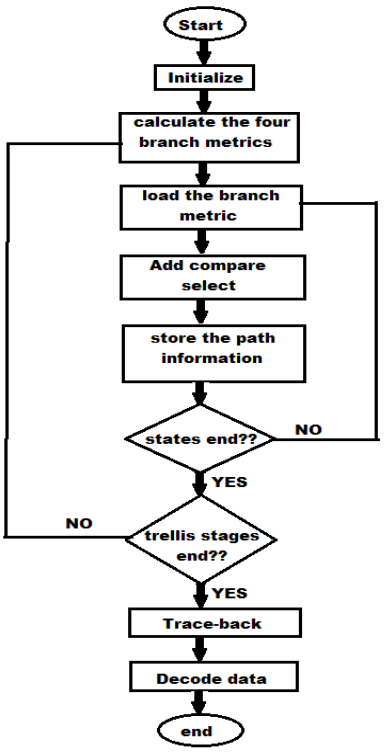

Fig.6 Flow Chart of Viterbi Trace back algorithm

Hardware and Software Requirement:

Software: Xilinx ISE, Xilinx VIVADO, Modelsim, MATLAB.
Language: Verilog HDL, MATLAB

- For Verilog HDL simulation
  Modelsim Simulator,
  Xilinx ISE, ISIM.
  For Synthesis, Hardware Generation
  Xilinx ISE 14.1
  Xilinx Vivado
  Supporting software's
  MATLAB 2013.(As a golden Reference)

IV. RESULTS AND INFERENCES

The Convolutional Encoder and Viterbi Decoder is designed and its behavioral simulation is verified using Xilinx. The simulation results are shown in Fig.7

After every seventh clock pulse, when the enable bit is high, the vit_data_out bits are equal to the bits inputted into the Convolutional encoder. In this manner the Viterbi decoder using trace-back approach, traces back the original input bit stream of data.

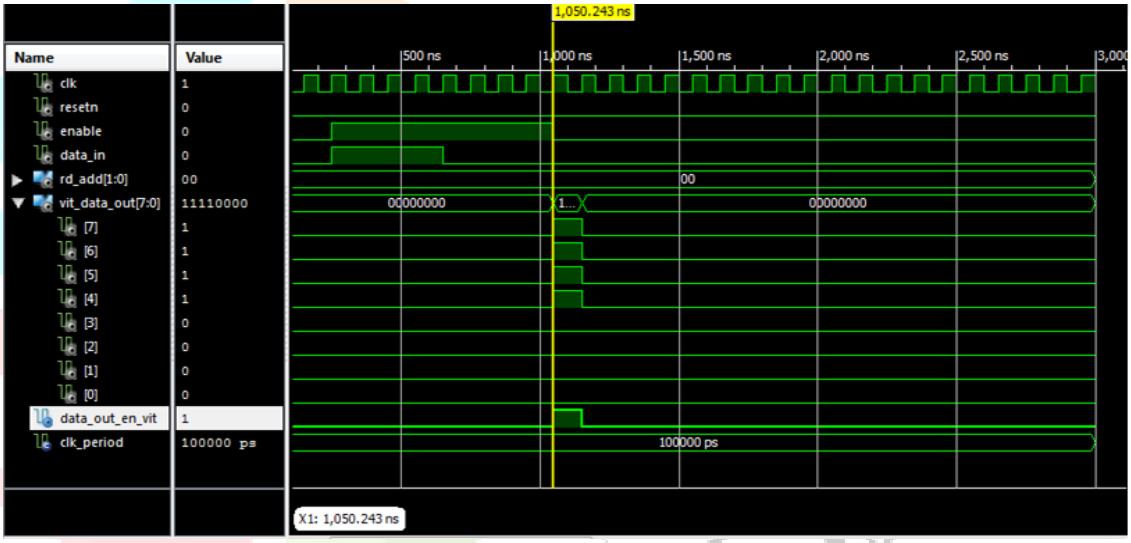

Fig.7 Simulation Results of Viterbi Decoder

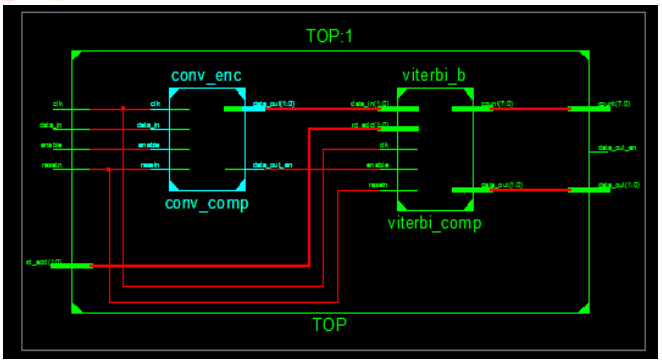

Fig.8 RTL Schematic of Convolutional Encoder and Viterbi Decoder

Timing Report for Existing Architecture:

```
Timing Summary:
---------------
Speed Grade: -5

   Minimum period: 34.663ns (Maximum Frequency: 28.849MHz)
   Minimum input arrival time before clock: 1.718ns
   Maximum output required time after clock: 5.248ns
   Maximum combinational path delay: No path found
```

Timing Report for proposed Architecture:

```
Timing Summary:
---------------
Speed Grade: -5

   Minimum period: 4.094ns (Maximum Frequency: 244.251MHz)
   Minimum input arrival time before clock: 3.663ns
   Maximum output required time after clock: 5.248ns
   Maximum combinational path delay: No path found
```

Device Utilization Summary for proposed architecture:

Table1.FPGA Design Utilization Summary of Viterbi Decoder for 1 bit error

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	358	704	50%
Number of Slice Flip Flops	33	1408	2%
Number of 4 input LUTs	641	1408	45%
Number of bonded IOBs	14	108	12%
Number of GCLKs	1	24	4%

V. CONCLUSION AND FUTURE SCOPE

In this paper, Convolutional Encoder and Viterbi Decoder (Hard Decision) is designed for SDR applications. The proposed architecture is pipelined ACSU and trace back method and to compute its performance characteristics in terms of device summary utilization, timing summary, delay of the circuit needs to be obtained and its simulation results observed and debugged. The Viterbi Decoder should be able to detect and correct one bit of error in the input bit stream of data. Further the proposed Architecture should work at a higher speed than the existing one. As it can be seen that the proposed architecture has better efficiency in terms of speed and device utilisation.it used a pipelined based architecture for increasing the frequency and finite state machine based decoder for efficient device utilisation.

Further the same work can be implemented using CADENCE or Synopsis tool, so the area and power consumption can be calculated. Interleaver can be added to accommodate burst errors.

REFERENCES

[1]A. Kumar, V. Ravichandran and S. Sudhakar, "A Novel Approach for FPGA Implementation of Register Exchange Based Viterbi Decoder and Re-Encoding based Node Synchronizer," 2019 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), GOA, India, 2019, pp. 1-6, doi: 10.1109/ANTS47819.2019.9118163.

[2]Z. Gao, J. Zhu, R. Han, Z. Xu, A. Ullah and P. Reviriego, "Design and Implementation of Configuration Memory SEU-Tolerant Viterbi Decoders in SRAM-Based FPGAs," in IEEE Transactions on Nanotechnology, vol. 18, pp. 691-699, 2019, doi: 10.1109/TNANO.2019.2925872.

[3]J. M. Ayarde, G. Tamagnone and G. Corral-Briones, "Timing Synchronization and Viterbi Decoding for FPGA-based SDR Platforms," 2019 XVIII Workshop on Information Processing and Control (RPIC), Bahía Blanca, Argentina, 2019, pp. 187-192, doi: 10.1109/RPIC.2019.8882145.

[4]J. M. Ayarde, G. Tamagnone and G. Corral-Briones, "Timing Synchronization and Viterbi Decoding for FPGA-based SDR Platforms," 2019 XVIII Workshop on Information Processing and Control (RPIC), Bahía Blanca, Argentina, 2019, pp. 187-192, doi: 10.1109/RPIC.2019.8882145.

[5]A. K. Kumar and P. S. Kumar, "High Speed Error-Detection and Correction Architectures for Viterbi Algorithm Implementation," 2019 3rd International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech), Kolkata, India, 2019, pp. 1-6, doi: 10.1109/IEMENTech48150.2019.8981127.

[6]M. Irfan, S. M. Shah and A. K. Khan, "Design and Implementation of Viterbi Encoding and Decoding Algorithm on FPGA," 2005 International Conference on Microelectronics, Islamabad, Pakistan, 2005, pp. 234-239, doi: 10.1109/ICM.2005.1590074.

[7] BupeshPandita ,Subir K Roy , “Design and Implementation of Viterbi Decoder using FPGAs”.