Implementation of Low-Latency Viterbi Architecture using with Encoded Operands

KONA.NARESH[1], RUPA KUMAR DANAVATH[2]
[1] *P.G Student, VLES*
[2] *Assistant Professor, Department of Electronics and Communication Engineering, Nagole Institute of technology and science*

ABSTRACT- The Viterbi algorithm is commonly applied to a number of sensitive usage models including decoding convolutional codes used in communications such as satellite communication, cellular relay, and wireless local area networks. Moreover, the algorithm has been applied to automatic speech recognition and storage devices. In this paper, efficient error detection schemes for architectures based on low-latency, low-complexity Viterbi decoders are presented. The merit of the proposed schemes is that reliability requirements, overhead tolerance, and performance degradation limits are embedded in the structures and can be adapted accordingly. We also present three variants of recomputing with encoded operands and its modifications to detect both transient and permanent faults, coupled with signature-based schemes. The instrumented decoder architecture has been subjected to extensive error detection assessments through simulations, and application specific integrated circuit (ASIC) [32 nm library] and field programmable gate array (FPGA) [Xilinx Virtex-6 family] implementations for benchmark. The proposed fine-grained approaches can be utilized based on reliability objectives and performance/implementation metrics degradation tolerance.

KEYWORDS- BMU,ACS,PCSA and FPGA

I. VITERBI ALGORITHM

1967 as a proficient technique for translating convolutional codes [1], generally utilized as a part of correspondence frameworks [2].This calculation is used for disentangling the codes utilized as a part of different applications including satellite correspondence, cell, and radio transfer. It has ended up being a successful answer for a ton of issues identified with computerized estimation. Additionally, the Viterbi decoder has down to earth use in usage of rapid (5 to 10 Gb/s) serializer-deserializers (SERDESs) which have basic idleness requirements. SERDESs can be additionally utilized as a part of neighborhood synchronous optical systems of 10 GB/s. Besides, they are utilized as a part of attractive or optical stockpiling frameworks, for example, hard plate drive or advanced video circle [3].

The Viterbi calculation process is like finding the in all likelihood arrangement of states, bringing about grouping of watched occasions and, in this way, gloats of high proficiency as it comprises of limited number of conceivable states [4– 7]. It is a viable usage of a discrete-time limited state Markov process apparent in memory less clamor and optimality can be accomplished by following the most extreme probability criteria [8]. It helps in following the stochastic procedure state utilizing an ideal recursive strategy which helps in the investigation and usage [9, 10].

A best level engineering for Viterbi decoders is appeared in Fig. 1.1. As found in this figure, Viterbi decoders are made out of three noteworthy segments: branch metric unit (BMU), include look at select (ACS) unit, and survivor way memory unit (SMU). BMU creates the measurements comparing to the paired trellis contingent upon the got flag, which is given as contribution to ACS which, at that point, refreshes the way measurements. The survival way is refreshed for every one of the states and is put away in the extra memory. SMU is in charge of dealing with the survival ways and giving out the decoded information as yield.

BMU and SMU units happen to be absolutely forward rationale. ACS recursion comprises of input circles; consequently, its speed is constrained by the cycle bound [11]. Consequently, the ACS unit turns into the speed bottleneck for the framework. M-step look-ahead procedure can be utilized to break the emphasis bound of the Viterbi decoder of limitation length K [12– 18]. A look-ahead strategy can consolidate a few trellis ventures into one trellis step, and if M > K, at that point throughput can be expanded by pipelining the ACS engineering, which helps in tackling the issue of cycle bound, and is much of the time utilized as a part of rapid correspondence frameworks.

Branch metric precomputation (BMP) which is in the front end of ACS is come about because of the look-ahead procedure and it commands the general intricacy and dormancy for profound look-ahead designs. BMP comprises of pipelined enrolls between each two successive advances and joins twofold trellis of various strides into a solitary complex trellis of one-advance. BMP rules the general idleness and unpredictability for profound look-ahead designs. Prior to the immersion of the trellis, just include activity is required. After the immersion of the trellis, include task is trailed by analyze activity where the parallel ways comprising of less measurements are disposed of as they are viewed as pointless.

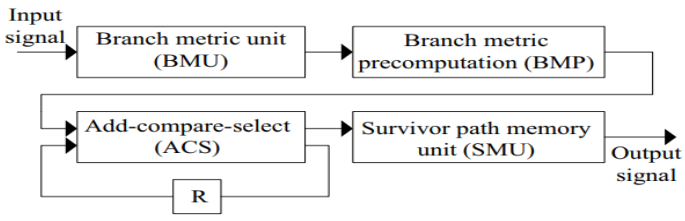

Fig.1: Viterbi decoder block diagram.

Despite the fact that Viterbi calculation structures are utilized normally in disentangling convolutional codes, within the sight of vast scale joining (VLSI) surrenders, wrong yields can happen which corrupt the exactness in unraveling of convolutional codes.

II. LITERATURE SURVEY

A. Binary Grouping (BBG) Approach

This area concentrates just on branch metric calculation, leaving aside the activities of look at and-dispose of. An ideal approach of BBG is contemplated with a specific end goal to expel all redundancies which are typically in charge of longer postponement and additional multifaceted nature, since different ways share regular calculations. Branch measurements calculation is said to be done consecutively for a traditional Viterbi decoder. At the point when two back to back paired trellis steps are consolidated, for each state, there are two approaching and two active branches, and the computational intricacy is 4 × N. As the outcomes don't rely upon the request of the trellis blend, the manner in which the trellis steps are assembled and consolidated aides in deciding the computational multifaceted nature. The mix in a retrogressive settled methodology can be clarified as takes after. The fundamental M-step trellises are partitioned into two gatherings comprising of m0 and m1 trellis steps. The twofold decay on every subgroup goes ahead till it turns into a solitary trellis step. The disintegration helps in evacuating most extreme conceivable repetition and, accordingly, accomplishes least deferral and multifaceted nature. At long last, it can be checked that the complexities engaged with the BBG approach are less when contrasted with the ones in the instinctive approach.

B. Look-ahead-based Low-Latency Architectures

This approach is an exceptionally effective plan approach in view of the BBG conspire for a general M which gives less or break even with inactivity, and furthermore has considerably less intricacy contrasted with other existing structures [3]. For imperative length K and M-step look-ahead, the execution of BMP is done in a layered way. A M-step trellis is a greater gathering comprising of MK sub-bunches with a trellis of K-step. In this manner, the aggregate quantities of P1 processors required are MK and each P1 is in charge of registering K-step trellises. Appropriately, we have the complexities and latencies of P1 and P2 as Comp.P1 = N(∑ki=2 2i) + N2, Comp.P2 = N2(N − 1) + N3, and Lat.P1,P2 = K, where N = 2k−1 is the quantity of trellis states. For P1 processors, the complex It y of include activity is N ∑k= 2i and that of the "think about" task is N2. So also, for P2 i2processors, the many-sided quality of include activity is N2(N −1) and that of the look at task is N3. For both P1 and P2 processors, the idleness is same, i.e., K; be that as it may, the multifaceted nature of P2 is bigger than that of P1. As the BBG approach is extremely proficient in registering the branch measurements, more activities of trellis blend can be allocated into BBG-based P1 processors with a specific end goal to lessen the quantity of P2 processors as they are costly as far as intricacy.

The trellis Steps L, which is figured in the P1 processors, has the requirement of being under 2 ×K so as to ensure that the dormancy include isn't lost. The quantity of gatherings Ng can be controlled by Ng = 2⌊log2(MK)⌋.

The general layered structure of the Viterbi calculation is appeared in Fig. 2.1 (in this figure, I, j ∈ [1, N] and l ∈ [1, K]). As found in this figure, inside two layers (appeared by Layer 1 and Layer 2 in Fig. 2.1), we have Ng steps, experiencing P1 and P2 processors. In every L-level P1 processor, the underlying advance mix is performed utilizing the BBG approach, trailed by linked add– look at tasks executed with extra special care for the rest of the L−K-step stage II calculation. In Layer 2, the yields of P1 processors are consolidated for registering. The last comparable complex trellis. This figure additionally demonstrates the P1 processor engineering in view of the BBG calculation. In Layer 1, despite the fact that P1 prompts longer dormancy, as the profundity of Layer 2 is diminished also, inertness punishment in not caused.

III. PROPOSED RELIABLE ARCHITECTURES

In this section, the error detection CSA and PCSA architectures are designed through recomputing with encoded operands, e.g., RERO, RESO, and variants of RESO, as shown in Figs. 3.1 and 3.3 with the locations of error detection modules shaded. Since this approach takes more number of cycles for completion, to alleviate the throughput degradation, the architecture is pipelined in the following fashion. First, pipeline registers are added to sub-pipeline the architectures, assisting in dividing the timing into sub-parts. The original operands are fed in during the first cycle. Nonetheless, during the second cycle, the second half of the circuit operates on the original operands and the first half is fed in with the rotated operands.

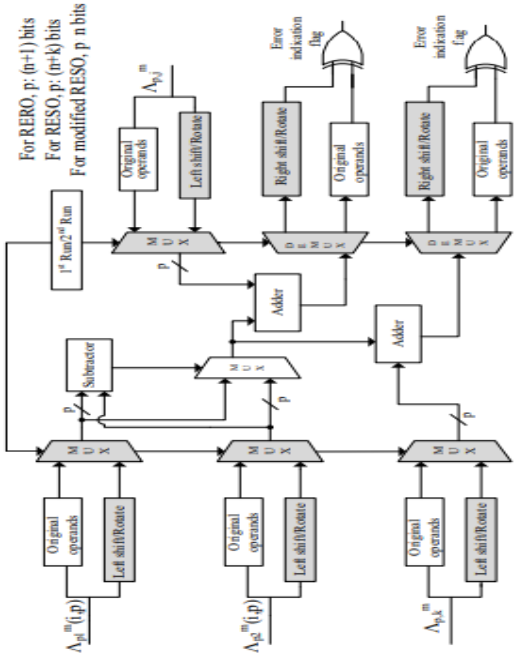

Fig.2: Recomputing with encoded operands for CSA.

For the CSA and PCSA architectures in Figs. 3.1 and 3.2, we also employ RESO and a RESO variant scheme for fault diagnosis. Both CSA and PCSA units consist of four inputs, each of them are passed in its original form and in the left shifted or rotated form to one of the multiplexers. If the select lines of these multiplexers are set to the first run, the original operands are passed without any change. If these are set to second run, the second (modified, i.e., left shifted/rotated) operands are passed. For the CSA unit, the inputs are fed to the subtractor and also to the multiplexer whose select line is set by the comparator. This serves as the design of compare-select unit. The output of the multiplexer is replicated and asserted as one of the inputs to two adders included in the design. The outputs of both of the adders are the outputs of the CSA unit. These are passed through the demultiplexers and the outputs of the demultiplexers are compared using an XOR gate, and the error indication flag is raised in case of an error. For the PCSA unit, the first two inputs are fed to the comparator which acts as the select line for the two multiplexers driven by the four adders used in the design. The other two inputs in combination with the previous inputs are given to the adders. The outputs of the two multiplexers are the outputs of the PCSA unit and to ensure that they are error-free, the outputs are passed through separate demultiplexers.

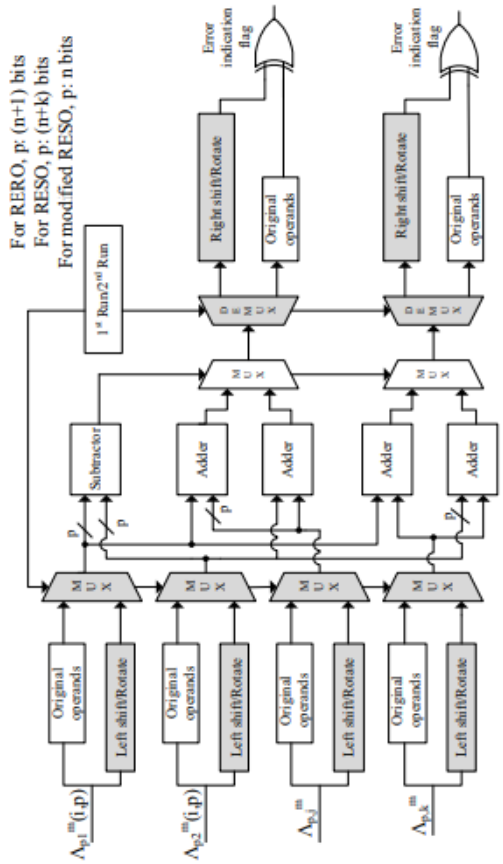

Fig.3: PCSA error detection through recomputing with encoded operands.

We have utilized RESO which performs the recomputation step with shifted operands, i.e., all operands are shifted left or right by k bits (this method is efficient in detecting k consecutive logic errors and k − 1 arithmetic errors). For CSA and PCSA architectures in Figs. 3.1 and 3.2, let us assume g(x, y) is the result of the operation which is stored in a register. The same operation is performed again with x and y shifted by certain number of bits. This new result g ′ (x, y) is stored and the original result g(x, y) can be obtained by shifting g ′ (x, y) in the opposite direction. Another used method in the proposed scheme is a modified version of the RESO scheme and this modification is that the bits that shift out are not preserved. This signifies that the total number of bits required for operation is only "n" bits and, hence, becomes more advantageous in terms of hardware cost than RESO and RERO methods, as pointed out in Figs. 3.1 and 3.2.

In modified RESO, only (n−k) LSBs of g(x) is compared with the shifted (n−k) LSBs of g ′ (x). This approach is a compromise between the area/power consumption and the error coverage. In order to execute the RERO method, we have added low hardware overhead to the initial design. RERO is used for detecting errors concurrently in the arithmetic units. Considering two n-bit rotations R and R −1 , suppose the input to an arithmetic function is x and g(x) is the output such that g(x) = R −1 × (g(R(x))). The result of g(x) computation happens to be the result of first run and R −1 × (g(R(x))) computation happens to be the second run. For both the CSA and PCSA units, we have used the RERO scheme in Figs. 3.1 and 3.2.

The first challenge in RERO for in Figs. 3.1 and 3.2 is to avoid the interaction between the MSB and LSB of the original operand during the recomputation operation. The second challenge in RERO for CSA and PCSA architectures is to ensure performance enhancements through sub-pipelining to increase the frequency and alleviate the throughput overhead as part of the FPGA and ASIC implementations. Finally, let us present a general approach for alleviating the throughput degradations of the proposed schemes. Suppose a number of pipeline registers have been placed to sub-pipeline the structures to break the timing path. Let us denote the n segments of the pipelined stages by Δ1- Δn. In a typical assertion, the original input can be first applied (to Δ1) and in the second cycle, while the second half (Δ2) of the architecture executes the first input, the encoded variant of the first input is fed. This trend can be scaled to n stages for normal (N) and encoded (E) operands.

IV. SIMULATION RESULTS

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	91	16640	0%
Number of Slice Flip Flops	112	33280	0%
Number of 4 input LUTs	150	33280	0%
Number of bonded IOBs	69	309	22%
Number of GCLKs	1	24	4%

Fig.4: DESIGN SUMMARY

```
================================================================================
Timing constraint: Default OFFSET OUT AFTER for Clock 'Clock'
  Total number of paths / destination ports: 6272 / 32
--------------------------------------------------------------------------------
Offset:              12.868ns (Levels of Logic = 6)
  Source:            OB_0 (FF)
  Destination:       YC1<7> (PAD)
  Source Clock:      Clock rising

  Data Path: OB_0 to YC1<7>
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FD:C->Q               5   0.591   0.776  OB_0 (OB_0)
     LUT2:I0->O            1   0.648   0.452  SC/Mxor_S0_0_xo<0>21 (N111)
     LUT4:I2->O            1   0.648   0.563  EAB/e026 (EAB/e026)
     LUT4:I0->O           36   0.648   1.406  EAB/e0206 (e0)
     LUT4:I0->O            8   0.648   0.900  SEC/Mmux_YC31211 (SEC/N2)
     LUT4:I0->O            1   0.648   0.420  SEC/Mmux_YC39 (YC3_2_OBUF)
     OBUF:I->O                 4.520          YC3_2_OBUF (YC3<2>)
    ----------------------------------------
    Total                     12.868ns (8.351ns logic, 4.517ns route)
                                       (64.9% logic, 35.1% route)

================================================================================
```

Fig.5: TIME SUMMARY

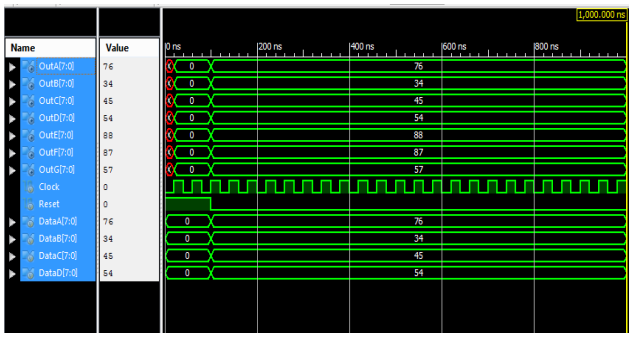

Fig.6: ENCODER OUTPUT

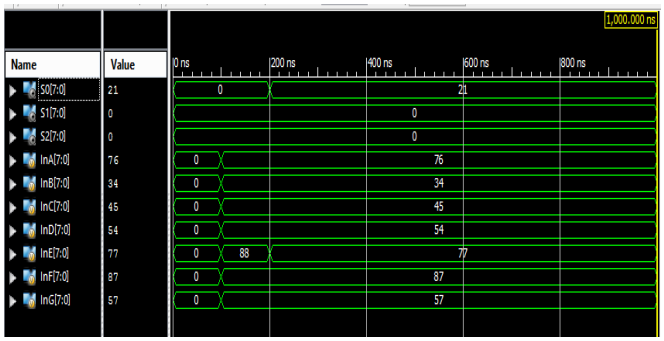

Fig.7: BMU OUTPUT

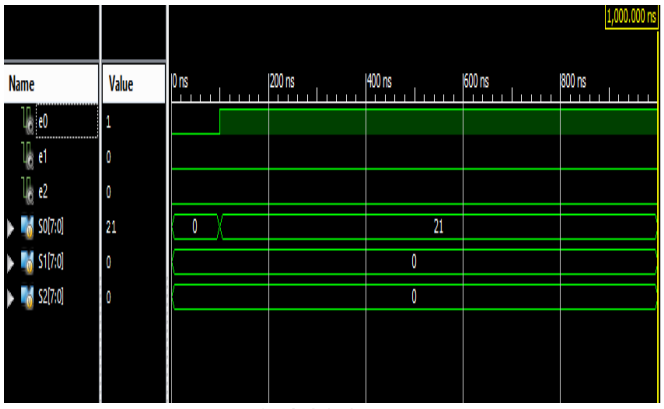

Fig.8: ACS OUTPUT

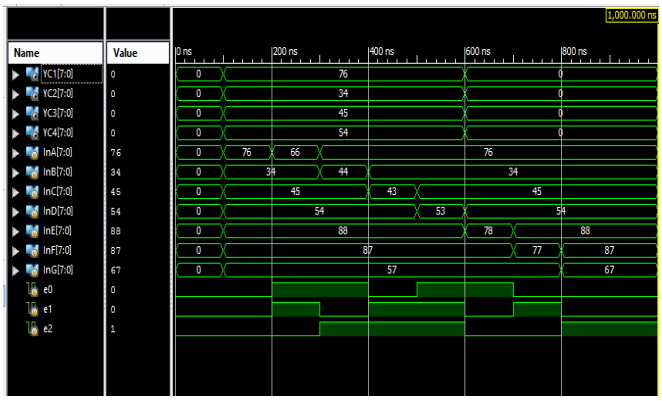

Fig.9: PCSA OUTPUT

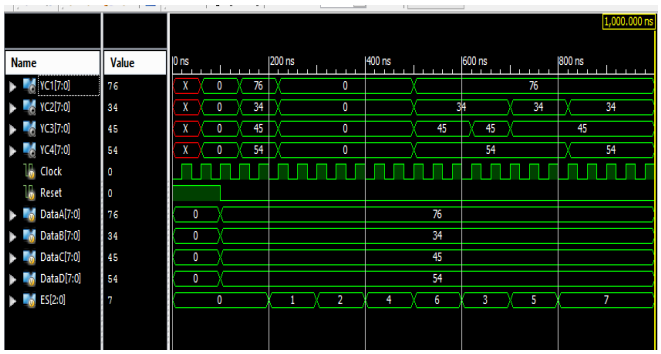

Fig.10: VITERBI OUTPUT

V. CONCLUSION

In this thesis, we presented fault diagnosis models for the CSA and PCSA units of low complexity and low-latency Viterbi decoder. The simulation results for the proposed methods of RESO, RERO, modified RESO, parity and self-checking adder based designs for both CSA and PCSA units show very high fault coverage (almost 100 percent) for the randomly distributed injected faults. The proposed architectures have been successfully implemented on Xilinx Virtex-6 Family and also by using the 32nm library using Synopsys Design Compiler for the ASIC implementation. Also, the ASIC and FPGA implementation results show that overheads obtained are acceptable. Thus the proposed models are reliable and efficient.

VI. REFERENCES

[1]. Massoud Pedram, "Power minimization in ic outline: Principles and applications," ACM Trans. Des. Autom. Electron. Syst., vol. 1, no. 1, pp. 3– 56, Jan. 1996.

[2]. Qing Wu, M. Pedram, and Xunwei Wu, "Clock-gating and its application to low power outline of successive circuits," Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on, vol. 47, no. 3, pp. 415– 420, Mar 2000.

[3]. G.E. Tellez, A. Farrahi, and M. Sarrafzadeh, "Action driven clock outline for low power circuits," in Computer-Aided Design, 1995. ICCAD-95. Process of Technical Papers., 1995 IEEE/ACM International Conference on, Nov 1995, pp. 62– 65.

[4]. E. Lee and A. Sangiovanni-Vincentelli, "Looking at models of calculation," in Proceedings of the 1996 IEEE/ACM global meeting on Computer-supported outline. IEEE Computer Society, 1997, pp. 234– 241.

[5]. Gilles Kahn, "The Semantics of Simple Language for Parallel Programming," in IFIP Congress, 1974, pp. 471– 475.
ssss
[6]. Edward A. Lee and David G. Messerschmitt, "Static booking of synchronous information stream programs for computerized flag handling," IEEE Trans. Comput., vol. 36, no. 1, pp. 24– 35, 1987.
[7]. E.A. Lee and T.M. Parks, "Dataflow process systems," Proceedings of the IEEE, vol. 83, no. 5, pp. 773 – 801, may 1995.
[8]. Syed Suhaib, Deepak Mathaikutty, and Sandeep Shukla, "Dataflow structures for GALS," Electronic Notes in Theoretical Computer Science, vol. 200, no. 1, pp. 33– 50, 2008.
[9]. Tzyh-Yung Wuu and Sarma B. K. Vrudhula, "Union of offbeat frameworks from information stream determination," Research Report ISI/RR-93-366, University of Southern California, Information Sciences Institute, Dec 1993.
[10]. Behnam Ghavami and Hossein Pedram, "Superior offbeat outline stream utilizing a novel static execution examination technique," Comput. Electr. Eng., vol. 35, no. 6, pp. 920– 941, Nov. 2009.
[11]. S.C. Brunet, E. Bezati, C. Alberti, M. Mattavelli, E. Amaldi, and J.W. Janneck, "Parceling and enhancement of abnormal state stream applications for multi clock space designs," in Signal Processing Systems (SiPS), 2013 IEEE Workshop on, Oct 2013, pp. 177– 182.
[12]. Simone Casale-Brunet, Analysis and enhancement of dynamic dataflow programs, Ph.D. proposition, STI, Lausanne, 2015.
[13]. Endri Bezati, High-level combination of dataflow programs for heterogeneous stages, Ph.D. postulation, STI, Lausanne, 2015.
[14]. S. Casale-Brunet, M. Mattavelli, and J.W. Janneck, "Cushion improvement in view of basic way investigation of a dataflow program configuration," in Circuits and Systems (ISCAS), 2013 IEEE International Symposium on, May 2013, pp. 1384– 1387.
[15]. Xilinx, Analysis of Power Savings from Intelligent Clock Gating, August 2012, XAPP790.