Alexandria Engineering Journal (2015) 54, 447–455

HOSTED BY

Alexandria University

Alexandria Engineering Journal

www.elsevier.com/locate/aej
www.sciencedirect.com

ORIGINAL ARTICLE

High performance ACS for Viterbi decoder using pipeline T-Algorithm

D. Vaithiyanathan [a,*], J. Nargis [b], R. Seshasayanan [a]

[a] Department of Electronics and Communication Engineering, Anna University, Chennai, India
[b] Department of Electronics and Communication Engineering, Aalim Muhammed Salegh College of Engineering, Chennai, India

Received 22 August 2014; revised 27 February 2015; accepted 15 April 2015
Available online 7 May 2015

KEYWORDS

Viterbi decoder;
T-Algorithm;
Add Compare Select (ACS);
Pipelining;
Field Programmable Gate Arrays (FPGAs)

Abstract Viterbi algorithm is the most popular algorithm used to decode the convolution code, but its computational complexity increases exponentially with the increasing constraint length due to the large number of Trellis transitions. However, high constraint length is necessary to improve the accuracy of the decoding process for the high rate convolution code. In particular, the Add Compare Select (ACS) module of the Viterbi Decoder will have large numbers of trellis states and trellis transitions with increased constraint lengths, which give rise to high hardware complexity and large power consumption. As the performance of Viterbi decoder mainly depends on its efficient implementation of ACS module, in this paper, we propose a modified pipelined architecture for the ACS of Viterbi decoder. This is derived by employing the technique of re-timing; further the architecture is also reconfigured to support various wireless standards. The architecture has been implemented in Xilinx Vertex 6 FPGA device to make the comparison between our architecture and the existing architecture. From the analysis done on ACS implementation, it is found that the resource requirements, delay and power consumption are optimized significantly for the proposed architecture compared to existing pipelined architecture. The results obtained from the analysis show that frequency of the system is increased up to 165 MHz with reduced area. The cell level performance is also obtained using Cadence Encounter (R) tool with TSMC 180 nm CMOS technology.

1. Introduction

Convolution codes are widely used in a variety of applications such as mobile communication, radio communication, and digital radio. Though convolution coding adopts a very simple procedure, decoding the code requires higher level of complex procedures. Hence Viterbi algorithm is adopted to decode the convolution codes. Thus the Viterbi decoder attains a high impact in decoding the convolution codes. For the convolution code with large constraint length, the hardware complexity and

* Corresponding author.
E-mail addresses: vaithi_d@rediffmail.com (D. Vaithiyanathan), nargisjaraab@gmail.com (J. Nargis), seshasayanan@annauniv.edu (R. Seshasayanan).

Peer review under responsibility of Faculty of Engineering, Alexandria University.

http://dx.doi.org/10.1016/j.aej.2015.04.007
1110-0168 © 2015 Faculty of Engineering, Alexandria University. Production and hosting by Elsevier B.V.

the power consumption of the Viterbi decoder are very high and this is the practical difficulty in the implementation of Viterbi decoder. More research works have been done on designing an area and power efficient Viterbi decoder to improve its efficiency and support for different applications [3–25].

Some specific techniques that the researchers have developed to reduce the computational load are M-Algorithm [1], T-Algorithm [2] [3], over scaling of the supply voltage [5]. M-Algorithm requires the sorting process for optimal path metric computation while the T-Algorithm needs to compute only the minimum value, so the T-Algorithm is found to be very much effective in achieving low power design. In the technique of supply voltage scaling, the entire system needs to be considered for the voltage scaling. According to T-Algorithm, the states which have the difference between optimal path metric and the corresponding state metric, lesser than the threshold would survive, while others would be discarded. By using this algorithm, power reduction can be achieved by reducing the number of states which are least likely to appear. But the optimal path has to be computed from the newly updated path metric and this limits the speed of ACS of Viterbi decoder. In [10,11,15,16] authors have contributed for the efficient implementation of T-Algorithm, but only the estimated optimal path metric is calculated. To overcome this problem, the technique incorporating the pre-computation architecture along with T-Algorithm is proposed by He et al. [17,18], in which optimal path metric is precomputed using previous path metric values and this results in improved clock speed. In our work we have analyzed the precomputation algorithm used in [17,18] and found a systematic way for further optimization by adopting some transformations on the location of delay elements.

Another perspective of this paper is the design of reconfigurable ACS architecture for the Viterbi decoder. Today's wireless devices are incorporated with many wireless standards. Each and every standard will have its own configuration parameters such as the constraint length and coding rate. This happens to be a major problem in the implementation of decoder within a single device. This leads to high cost design and large silicon area requirement. To overcome this problem, we need a flexible reconfigurable architecture which should adapt to different wireless standards of different configuration parameters. In this work, we proposed the high performance ACS architecture for Viterbi decoder. The advantages of the proposed techniques are validated with the FPGA implementation results and are reported for both existing and proposed techniques.

The rest of this paper is organized as follows. In Section 2, the theoretical background information of the Viterbi decoder is given. Section 3 provides the ACS design for rate 3/4 convolution code based on precomputation architecture along with T-Algorithm. Section 4 describes the proposed ACS architecture of the Viterbi decoder. Comparison of the results taken in the existing and proposed technique is discussed in Section 5. Finally our work is concluded in Section 6.

2. Background

2.1. Functional description of Viterbi decoder

The process of convolutional encoding the data is achieved using a cascaded connection of flip flops and associated combinatorial logic that performs modulo-two addition. The encoder used for analysis is rate 3/4 convolution encoder with the constraint length of 7 is shown in Fig. 1. This encoder is chosen to illustrate the effectiveness of the proposed architecture for such high rate codes. When a convolutional encoded symbol is transmitted over the channel, it might get corrupted by noise. To recover the data from such a corrupted symbol, Viterbi decoder is used, whose functional block diagram is shown in Fig. 2 that comprises of Branch Metric Unit (BMU), Add-Compare-Select Unit (ACSU), Path Metric Unit (PMU) and Survivor Memory Unit (SMU). The branch metric unit's function is to calculate branch metrics, which are normed distances between the ideal code word and the received symbol. The Add Compare Select Unit (ACSU) adds the BMs to the corresponding previous PMs, compares the new PMs, and then stores the survivor PMs in the Path Metric Memory. At the same time, the ACSU stores the associated survivor path decisions in the Survivor Memory Unit

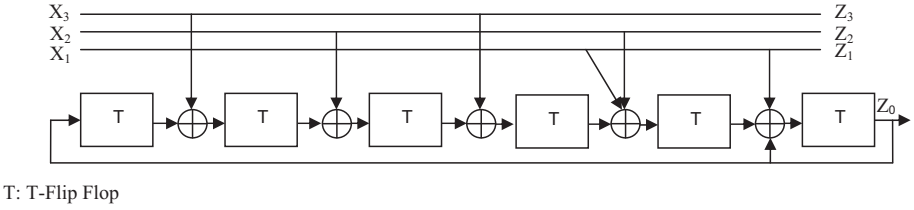

Figure 1 Rate 3/4 convolution encoder.

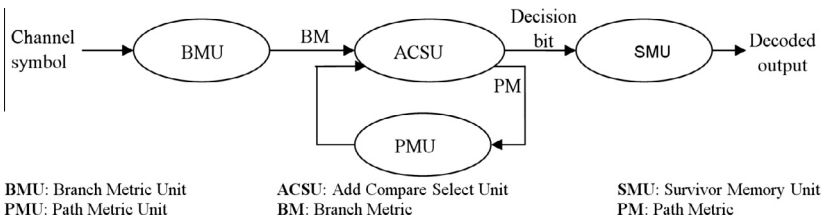

Figure 2 Functional block diagram of Viterbi decoder.

(SMU), and then either Register exchange mechanism or Trace back Algorithm is employed to the SMU to regenerate the decoded sequence.

2.2. Problem description of Viterbi decoder

To prove the fact that the computational complexity of Viterbi Decoder is high for large constraint length convolution codes, comparison is made between the low rate 1/2 code and the high rate 3/4 code and it is discussed below. Rate 1/2 convolution encoder with constraint length as 3 has four states and two incoming paths for every node. Thus the rate 1/2 coder has totally 8 (4×2) paths in every ACSU cycle. Rate 3/4 convolution encoder with constraint length as 7 has sixty-four states and eight incoming paths for every node. Therefore rate 3/4 coder contains 512(64×8) paths in every ACSU cycle. This demonstrates that the high rate convolution code has large amount of computation which leads to high hardware complexity, more delay and large power consumption. In order to reduce the computational complexity and high power consumption, we proposed a modified pipelined version of ACSU architecture.

3. T-Algorithm with 2-step precomputation architecture

3.1. T-Algorithm

In high rate convolution codes, the probability of transitions is very high which gives rise to the large dynamic power dissipation. To solve this problem, the number of transitions has to be reduced which can be done by reducing the number of states. This principle is the basis of T-Algorithm [17] in which the least likely states are discarded; those states cannot participate in next cycle computations. To find the least likely states, this algorithm compares the difference between the state metric and the optimal path metric of every state with the threshold, states whose difference greater than the threshold are considered as the least likely states. The straight forward

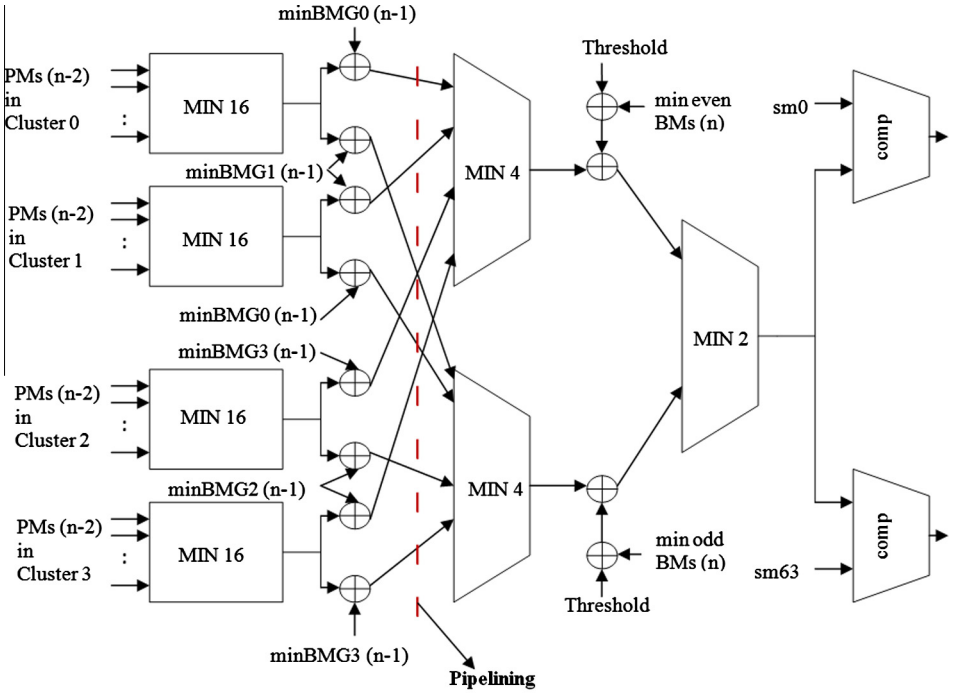

Figure 3 Architecture of 2-step pre-computation T-Algorithm for rate 3/4 convolution code.

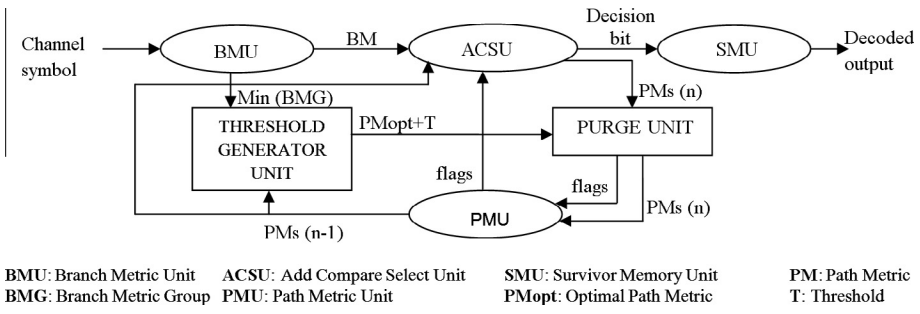

Figure 4 Functional block diagram of Viterbi decoder with pre-computation architecture.

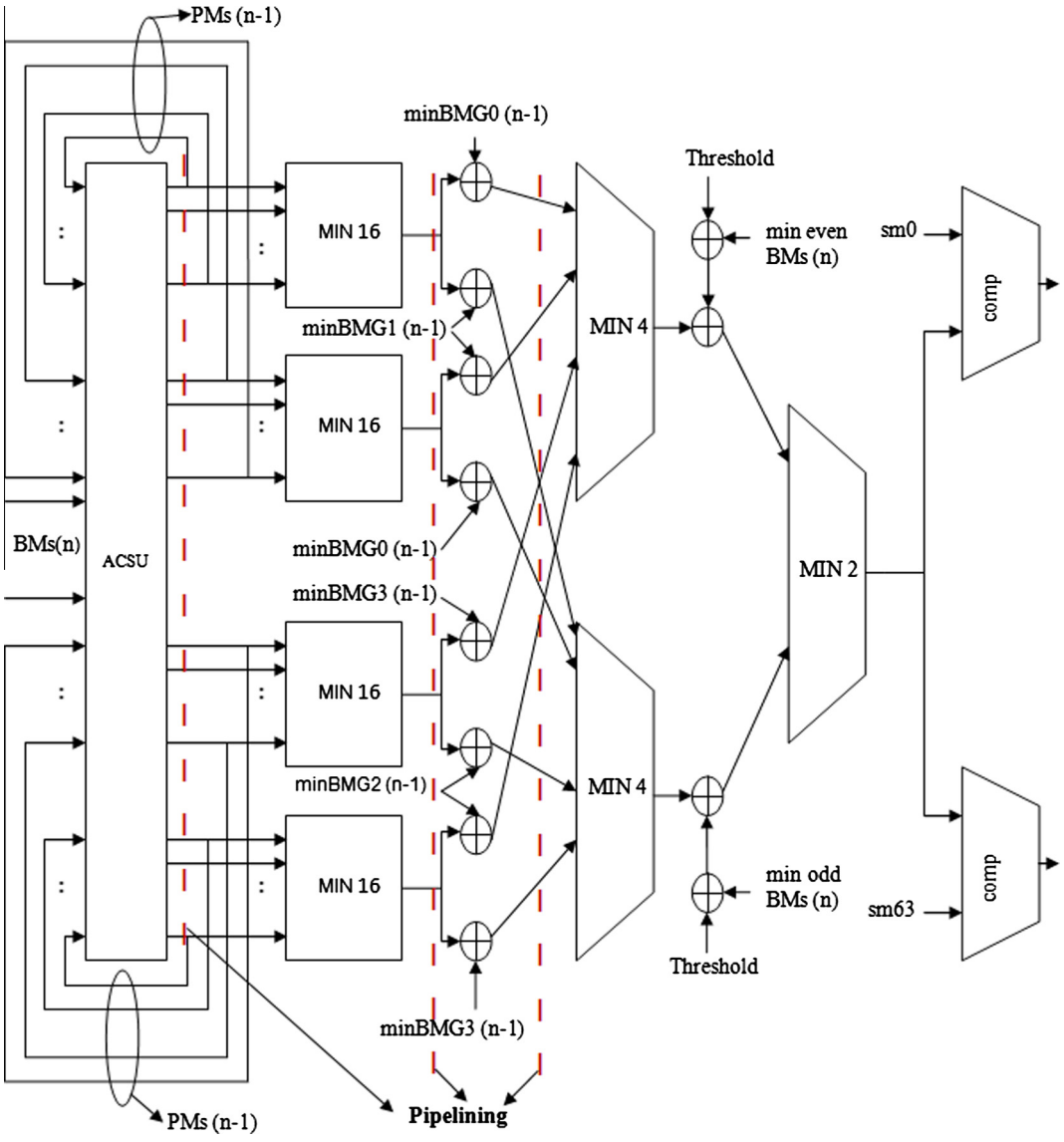

Figure 5 Proposed pipelined architecture for rate 3/4 convolution code.

implementation of T-Algorithm requires three stages of 4-input comparator, one adder for branch metric computation, one 4-input comparator and one 2-input comparator for the calculation of best path metric from 8 incoming path metric, one adder for adding the threshold with the optimal path metric, one 2-input comparator for the purge unit. The critical path achieved from this conventional Viterbi decoder is expressed as Eq. (1),

$$T = 2T_{adder} + 4T_{4-i/p\_comp} + 2T_{2-i/p\_comp} \quad (1)$$

From Eq. (1), it is observed that the critical path time of the conventional implementation of T-Algorithm is very high which limits the speed of the system.

3.2. T-Algorithm with precomputation architecture

To overcome the limitation of the straight forward implementation of T-Algorithm, T-Algorithm with Pre-computation architecture for rate 3/4 convolution code is proposed in [17,18] which is shown in Fig 3. In this method, optimal path metric is calculated as a function of previous path metric PMs $(n-q)$, where 'q' is the pre-computation step.

Accurate path metric is calculated in every cycle. So, no extra parameters are required for compensation. Since the optimal path metric is calculated from the pre-computed values, the long critical path is shortened for this architecture. Functional block diagram of Viterbi decoder with this architecture is shown in Fig. 4. For two step pre-computation architecture, optimal path metric can be calculated within 2 clock cycles. So, it can be pipelined as two stages. The critical path time of two stages is expressed as Eqs. (2) and (3),

$$T_{(stage\_1)} = T_{adder} + 2T_{4-i/p\_comp} \quad (2)$$

$$T_{(stage\_2)} = T_{adder} + T_{4-i/p\_comp} + 2T_{2-i/p\_comp} \quad (3)$$

From this, it is observed that the path time of either of these stages is less compared to the one given in Eq. (1).

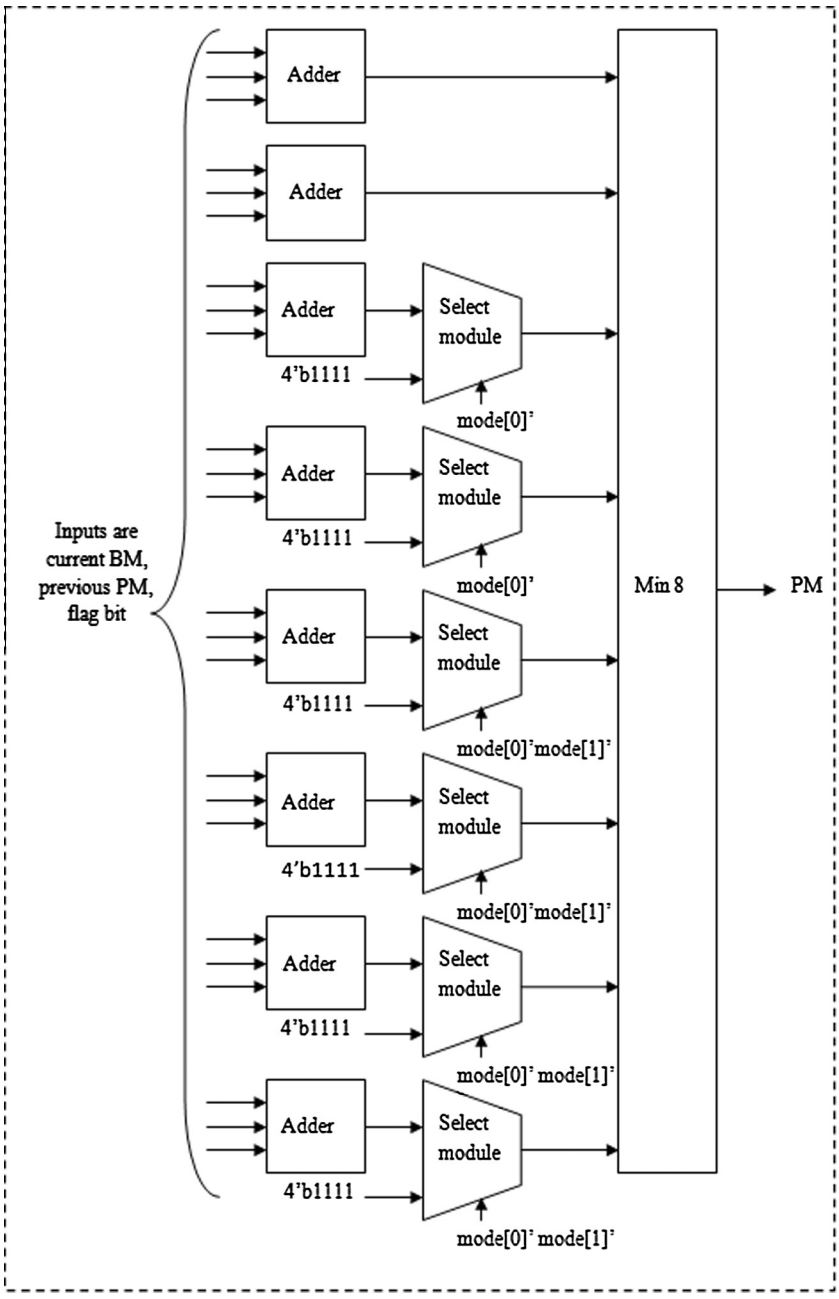

Figure 6 Single ACS module for reconfigurable ACS architecture.

The details on BG group, minBMG, even and odd BMs are given in [18].

4. Proposed ACSU architecture of Viterbi decoder

In the existing technique [18] which is discussed in the previous section, all the 64 path metrics are stored in the registers for the successive two clock cycles. After this process, the stored path metric values are grouped into four clusters, and then the minimum path metric value of each cluster is determined. Based on these computations, optimal path metric is calculated which is understood from Fig. 3. In our approach, the locations of delay elements are changed without affecting the functionality of the system. Here the minimum value finder block of each cluster is moved before one set of delay elements i.e. minimum value of each cluster is computed in terms of PMs $(n-1)$ instead of PMs $(n-2)$. Then the minimum value of each cluster is maintained for one cycle. By doing this kind of modification, three stage pipelined circuit is obtained without increasing the count of delay elements, whereas for the existing system, we have only two stages of pipelining. Pipelined three stages are shown as dotted lines in Fig. 5. The critical path time of the three stages is expressed as Eqs. (4)–(6).

$$T_{(stage\_1)} = 2T_{4-i/p\_comp} \tag{4}$$

$$T_{(stage\_2)} = T_{adder} \tag{5}$$

$$T_{(stage\_3)} = T_{adder} + T_{4-i/p\_comp} + 2T_{2-i/p\_comp} \quad (6)$$

Since only the minimum value is retained for the consequent cycle, the number of registers is reduced to a great extent. The critical path of the first stage of the existing system contains one adder and two no.'s 4-input comparator which involves more delay, whereas the proposed technique is divided into two stages to obtain reduced delay along this path. Even though the critical path time of the third stage of our system is same as the second stage of the existing system, our method achieves better delay reduction as it has lesser number of registers and look-up tables (LUT's). Therefore more freedom in placement and routing of the instances is gained and also by giving proper area and timing constraints in the Xilinx tool, the delay can be controlled much more effectively for the proposed method compared to the existing system. Also the switching activity of the system comes down due to the decrease in the register count which results in reduced dynamic power dissipation.

4.1. Reconfigurable ACS architecture

In the process of designing the reconfigurable ACS architecture the similarities between different parameters of the convolutional codes are first found out. From this, the basic cell of the design is constructed and then the control logic is added to control different parameters of the convolutional code. In our design, highest configuration parameter is rate 3/4 convolution code with constraint length 7. To decode the rate 3/4 convolution encoder, 64 ACS units are required, and the structure of single ACS unit is given in Fig. 6, which comprises of eight adders. These adders will get enabled depending upon the flag condition, if the required mode is rate 3/4 convolution coder, all eight adders in the ACS unit will get enabled depending on the flag condition. If the mode required is rate 2/3 convolution coder, only the top four adders of the ACS unit will be activated, and the rest four adders will be disengaged. For the case of rate 1/2 convolution encoder, only the top two adders are required. Since for all these three cases, top two adders are required, the output from these adders is directly given to the min 8 comparator. Other adders are controlled by having the select module logic. If the mode input is 00, the design will take all adders from the ACS module which is given in Fig. 6. If the mode is 01, the design will take four adders required for rate 2/3 convolutional encoder. If the mode is 10, the design will take only two adders required for rate 1/2 convolution encoder.

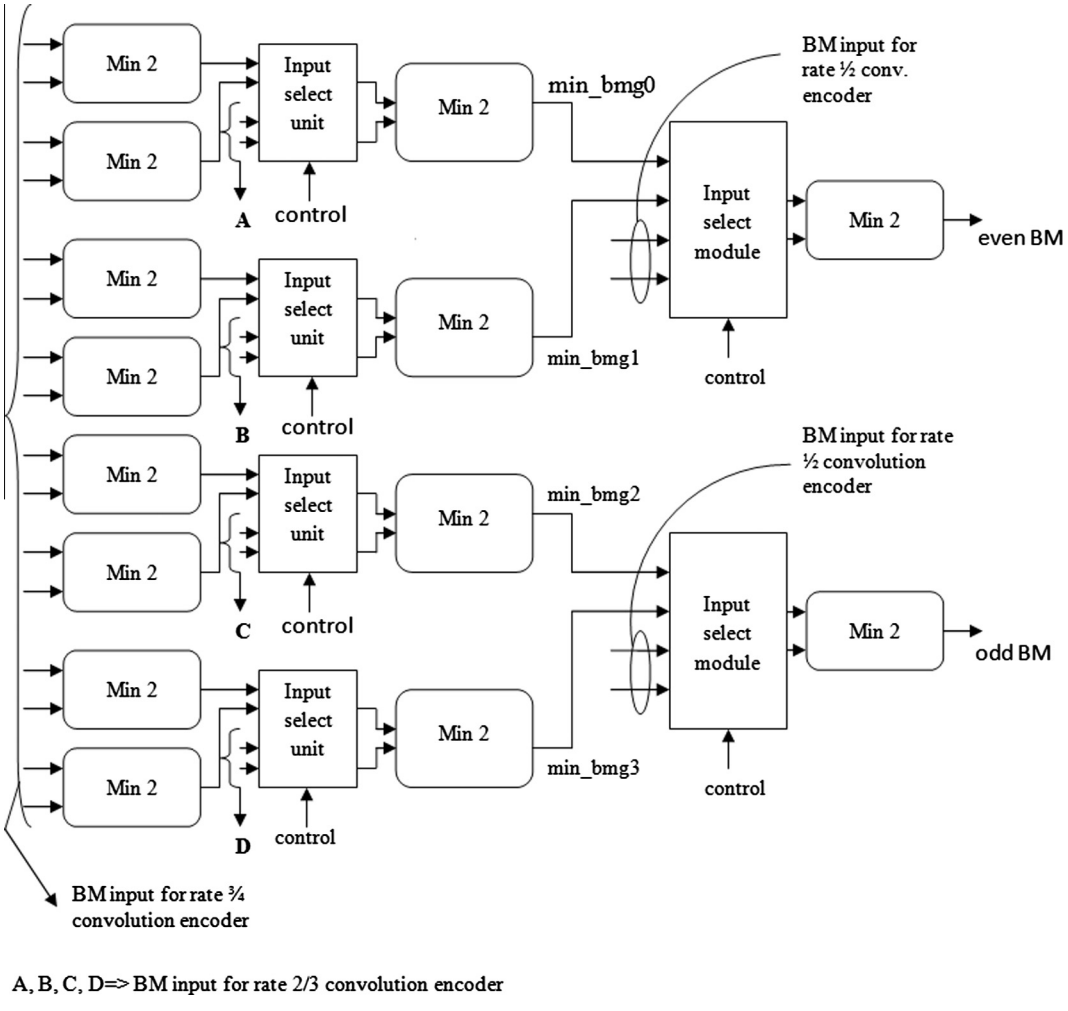

Figure 7 minBMG, oddBM, evenBM computation for reconfigurable ACS architecture.

For optimal path metric computation, it is required to find the minBMG, oddBM, evenBM values whose functional diagram is shown in Fig. 7. Since the coder has four branch metric values in one cluster, rate 3/4 convolution coder requires two stages of two input comparator. Rate 2/3 convolution coder has only two branch metric values, so it requires only one stage of two input comparator. To facilitate the selection between rate 3/4 and rate 2/3 mode, input select unit is inserted between the comparator stages. Depending upon the input mode, the input select unit will accept the inputs of the required mode and it will feed the accepted values to the rest of the computation. For rate 1/2 convolution encoder, comparators are not required for minBMG computation. Only evenBM, oddBM computation requires one comparator. To enable this selection, another input select unit is added.

From this architecture, it is clear that for the highest parameter mode all the blocks are utilized. For others, a portion of hardware is utilized. The resources are shared and reused among different modes. The proposed reconfigurable decoder supports the coding rate 1/2, 2/3 and 3/4 with constraint length 7. The reconfigurable architecture is implemented with negligible hardware overhead compared to the design with separate circuitry.

5. Experimental results and comparisons

In order to validate the performance of the proposed system, the architecture of both existing [18] and proposed ACS are described in Verilog HDL and targeted to Xilinx Vertex 6 XC6VLX240T-1FF1156 device [26]. The existing and the proposed architectures are compared in terms of hardware complexity, timing delay and power consumption. Parameters for the convolution encoder are specified as follows. Constraint length is chosen as 7 with coding rate as 3/4 and number of states is 64. The received sequence is examined by considering 4-bit word per cycle. We have observed that the area of ACS of Viterbi decoder is dominated by the registers. This makes the ACS expensive in terms of area and this large no's of registers give rise to high power consumption. But in the proposed method, since we have calculated the minimum cluster value using PMs $(n-1)$ instead of PMs $(n-2)$, the number of registers and LUT is highly reduced. Resource requirements of both proposed and existing architectures are summarized in Table 1. It is observed from Table 1, the number of slice registers and LUT'S has decreased by about 39% and 6% respectively for the proposed architecture compared to the He et al. [18]. From this it is very clear that the proposed architecture is very efficient compared to the existing one in terms of hardware complexity.

Timing result is also noted for the proposed and existing system and is reported in Table 1. To get the accurate timing result, Post Place and Route (PAR) is done. Since the hardware resource requirements become low for the proposed method, it gains greater flexibility in placement and routing to get the optimized delay. We compared the power consumption of the existing architecture with our modified pipelined architecture using Xilinx XPower analyzer tool. Reduced register count in our architecture decreases the switching rate which in turn reduces the dynamic power dissipation. For power estimation, the clock frequency was set to 165 MHz and supply voltage of Vccint = 1.5 V. Information obtained on chip power and supply power is reported in Table 2. Furthermore, the proposed and existing designs are synthesized using Cadence® RTL compiler® with standard library of TSMC 180 nm CMOS technology [27], and the hardware complexity in terms of cell area, power consumed and delay calculations are summarized in Table 3. It is observed from Table 3 that the proposed design consumes less area and 18.7% reduction in power consumption with 6.5% reduction in total delay compared to the existing architecture.

The design of ACS unit is constructed as a re-configurable one to achieve efficient resource reuse and to allow the design to occupy a relatively smaller chip area. The resource utilization summary of the reconfigurable ACS is summarized in Table 4. The proposed reconfigurable ACS for the constraint length 7 with coding rate 1/2, 2/3 and 3/4 has significant

Table 1 Resource utilization and timing results.

Logic utilization	He et al. [18]	Proposed system
Slice register count	636	387
Slice LUTs count	6681	6271
Post PAR timing results	6.436 ns	6.085 ns

Table 3 Cell level analysis report.

Parameter	He et al. [18]	Proposed system
Cell count	15,775	15,510
Cell area	64,644	61,387
Power (nW)	1265295.849	1028261.037
Delay (ps)	473	442

Table 4 Resource utilization for reconfigurable ACS.

Logic utilization	Proposed system	Proposed reconfigurable system
Slice register count	387	432
Slice LUTs count	6271	8267
Post PAR timing results for reconfigurable ACS	6.085 ns	6.257 ns

Table 2 Power report summary.

Technique	On chip power summary (W)						Supply power summary (W)		
	Clock	Logic	Signal	IO	Leakage	Total	Dynamic	Quiescent	Total
He et al. [18]	0.031	0.074	0.104	0.012	1.972	2.193	0.221	2.761	2.982
Proposed	0.024	0.077	0.093	0.012	1.971	2.177	0.206	2.761	2.966

Table 5 Power report summary for the reconfigurable ACS.

Technique	On chip power summary (W)						Supply power summary (W)		
	Clock	Logic	Signal	IO	Leakage	Total	Dynamic	Quiescent	Total
Proposed ACS	0.024	0.077	0.093	0.012	1.971	2.177	0.206	2.761	2.966
Proposed reconfigurable ACS	0.028	0.115	0.245	0.026	1.978	2.393	0.415	2.767	3.182

Table 6 Comparison with reported architectures on xilinx xcv800 FPGA.

Architecture	Area (gates)	Throughput (Mbps)
Proposed structure	57,927	35.876
Structure of [18]	62,637	35.458
Structure of [19]	108,389	35.558
Structure of [20]	86,845	20

Table 7 Comparison of reconfigurable Viterbi decoder with reported architectures on xilinx xc2vp30 FPGA.

Architecture	Constraint length supported	Area (gates) (K)	Throughput	Power (mw)
Proposed structure	7	66.828	80.142 Mbps	103
Structure of [19]	3–7	113	81 Mbps	103
Structure of [20]	3–7	89.5	20 Mbps	–
Structure of [21]	3–7	175	70 Mbps	–
Structure of [22]	7,9	95	12.5 Mbps	–
Structure of [23]	3–9	190	60.5 Mbps	561.64
Structure of [24]	4–9	65	333.7 Kbps	–

hardware overhead compared to the proposed nonreconfigurable ACS. If we have separate circuitry for these configuration parameters on the single chipset, the requirement will be huge. So, in this perspective the hardware overhead of the reconfigurable ACS compared to the nonreconfigurable ACS is negligible. PAR timing results and power report are also summarized in Tables 4 and 5 respectively. The maximum frequency achieved for this reconfigurable Viterbi decoder is 160 MHz. Power consumption of the reconfigurable ACS is almost same as the nonreconfigurable ACS. Thus the resulting reconfigurable ACS architecture is found to be efficient in terms of area and has almost the same timing and power results as the nonreconfigurable ACS architecture.

To compare with the reported architectures presented in [19–24], the proposed architecture and architecture in [18] were also tested on Xilinx XCV800 FGPA and XC2VP30 FPGA. Table 6 compares the proposed decoder and [18] for fixed constraint length 7 and code rate as 3/4 with the other existing architecture that has the constraint length of 7 with code rate code rate as 1/2–1/3 presented in [19,20]. From Table 6, it is understandable that the proposed structure has lesser area utilization of 7.5%, 46.56% and 33.29% compared to the structure in [18–20] respectively. At the same time the proposed architecture maintains its throughput compared to [18,19] and 44.25% higher than the reported [20].

In order to obtain more realistic results, Table 7 compares the proposed reconfigurable decoder which supports the coding rate 1/2, 2/3 and 3/4 with constraint length 7 with the reported architecture presented in [19–24]. When compared to the most recent one [19], the proposed architecture consumes 40.86% lesser area and almost maintains the same throughput. The structure reported in [20], consumes 25.33% more area and is 300% slower than the proposed. Compared to [21], the proposed architecture has 61.81% area improvement and 14.48% speed improvement. The logic utilization by [24] is significantly lower than the other structures, but throughput is very low. The overall results show that, the proposed scheme utilizes less logic resources with high throughput and is found to be efficient for high rate convolution code.

6. Conclusion

In this paper, we have presented a modified approach for the ACS design of Viterbi decoder, by changing the locations of delay elements in an optimized manner without affecting the functionality and thus results a good reduction in register count and critical path time. To improve the flexibility of ACS design and for efficient hardware reuse of various wireless standards, the proposed ACS design is reconfigured for different configuration parameters. The ACS design is reconfigured with negligible hardware overhead compared to the fixed implementation of Viterbi decoder. The modified design is successfully implemented on the Xilinx Vertex 6 FPGA device. This method has decreased the number of registers by 39% and LUT count by 6%. The operating frequency of the system is increased up to 165 MHz. Furthermore, to get in-depth information to prove the efficiency of the proposed technique, the design is also synthesized using Cadence® RTL compiler® using TSMC 180 nm CMOS library to obtain cell level performance parameters. Thus the proposed architecture achieves high-speed and low-power and is found to be very effective for the high rate convolution code.

References

[1] C.F. Lin, J.B. Anderson, M-Algorithm Decoding of Channel Convolutional Codes, Princeton Conf. Info. Sci. Syst., Princeton, NJ, 1986.

[2] S.J. Simmons, Breadth-first trellis decoding with adaptive effort, IEEE Trans. Commun. 38 (1) (1990) 3–12, http://dx.doi.org/10.1109/26.46522.

[3] F. Chan, D. Haccoun, Adaptive Viterbi decoding of convolutional codes over memoryless channels, IEEE Trans. Commun. 45 (11) (1997) 1389–1400, http://dx.doi.org/10.1109/26.649755.

[4] S.W. Choi, K.M. Kang, S.S. Choi, A two-stage Radix-4 Viterbi decoder for multiband OFDM UWB system, ETRI J. 30 (6) (2008) 850–852.

[5] R.A. Abdallah, N.R. Shanbhag, Error-resilient low-power Viterbi decoder architectures, IEEE Trans. Signal Proc. 57 (12) (2009) 4906–4917, http://dx.doi.org/10.1109/TSP.2009.2026078.
[6] M. Guo, M.O. Ahmad, M.N.S. Swamy, C. Wang, A low-power systolic array-based adaptive Viterbi decoder and its FPGA implementation, in: Proceeding of the ISCAS 2003, 276–279 May 2003, http://dx.doi.org/10.1109/ISCAS.2003.1205960.
[7] M. Guo, M.O. Ahmad, M.N.S. Swamy, C. Wang, FPGA design and implementation of a low-power systolic array-based adaptive Viterbi decoder, IEEE Trans. Circuits Syst. I 52 (2) (2005) 350–365, http://dx.doi.org/10.1109/TCSI.2004.838266.
[8] R. Tessier, S. Swaminathan, R. Ramaswamy, D. Goeckel, W. Burleson, A reconfigurable, power-efficient adaptive Viterbi decoder, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 13 (4) (2005) 484–488, http://dx.doi.org/10.1109/TVLSI.2004.842930.
[9] F. Angarita, A. Perez-Pascual, T. Sansaloni, J. Valls, Efficient mapping on FPGA of a Viterbi decoder for wireless LANs, in: Proceeding of the Workshop on Signal Processing Systems Design and Implementation, 710–7155 November 2005, http://dx.doi.org/10.1109/SIPS.2005.1579957.
[10] M.-H. Chan, W.-T. Lee, M.-C. Lin, L.-G. Chen, IC design of an adaptive Viterbi decoder, IEEE Trans. Consum. Elect. 42 (1996) 52–62, http://dx.doi.org/10.1109/30.485461.
[11] Fei Sun, Tong Zhang, Low power state-parallel relaxed adaptive Viterbi decoder design and implementation, IEEE ISCAS (2006) 4811–4814, http://dx.doi.org/10.1109/ISCAS.2006.1693707.
[12] S.W. Shaker, S.H. Elramly, K.A. Shehata, FPGA Implementation of a reconfigurable Viterbi decoder for WiMAX receiver, in: Proceeding of the Inter. Conf. on microelectronics, 264–267 December 2009, http://dx.doi.org/10.1109/ICM.2009.5418636.
[13] L. Bissi, P. Placidi, G. Baruffa, A. Scorzoni, A Viterbi decoder architecture for a standard-agile and reprogrammable transceiver, Integration, VLSI J. 41 (2) (2008) 161–170, http://dx.doi.org/10.1016/j.vlsi.2007.04.001.
[14] J.-S. Han, T.-J. Kim, C. Lee, High performance Viterbi decoder using modified register exchange methods, in: Proc. of the International Symposium on Circuits and Systems, vol. 3, 2004, pp. 553–556, http://dx.doi.org/10.1109/ISCAS.2004.1328806.
[15] Fei Sun, Tong Zhang, Parallel high-throughput limited search trellis decoder VLSI design, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 13 (9) (2005) 1013–1202, http://dx.doi.org/10.1109/TVLSI.2005.857181.
[16] J. Jin, C.-Y. Tsui, Low-power limited-search parallel state Viterbi decoder implementation based on scarce state transition, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 15 (11) (2007) 1172–1176, http://dx.doi.org/10.1109/TVLSI.2007.903957.
[17] J. He, H. Liu, Z. Wang, A fast ACSU architecture for Viterbi decoder using T-Algorithm, in: Proc. Of the 43rd IEEE Asilomar Conf. Signals, Syst. Comput., November 2009, pp. 231–235, http://dx.doi.org/10.1109/ACSSC.2009.5470119.
[18] Jinjin He, Huaping Liu, Zhongfeng Wang, Xinming Huang, Kai Zhang, High-speed low-power Viterbi decoder design for TCM decoders, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 20 (4) (2012) 755–759, http://dx.doi.org/10.1109/TVLSI.2011.2111392.
[19] C. Vennila, A.K. Patel, G. Lakshminarayanan, S.B. Ko, Dynamic partial reconfigurable Viterbi decoder for wireless standards, Comput. Electr. Eng. 39 (2) (2013) 164–174, http://dx.doi.org/10.1016/j.compeleceng.2012.12.009.
[20] K. Chadha, J. Cavallaro, A reconfigurable Viterbi decoder architecture, in: Proc. of the Thirty Fifth Asilomar Conference on Signals, Systems and Computer, vol. 1, November 2001, pp. 66–71, http://dx.doi.org/10.1109/ACSSC.2001.986882.
[21] J.M. Campos, R.A. Cumplido, A runtime reconfigurable architecture for viterbi decoding, in: Proc. of the 3rd International Conference on Electrical and Electronics Engineering, 1–4 September 2006, http://dx.doi.org/10.1109/ICEEE.2006.251908.
[22] R. Rasheed, A. Menouni, R. Pacalet, Reconfigurable Viterbi decoder for mobile platform, in: Proc. of the IFIP International Conference on Mobile and Wireless Communication Networks, 19–21 September 2005.
[23] J.R. Cavallaro, M. Vaya, Viturbo: a reconfigurable architecture for viterbi and turbo decoding, in: Proc. of the International Conference on Acoustics, Speech and Signal Processing, vol. 2, April 2003, pp. 1520–6149, http://dx.doi.org/10.1109/ICASSP.2003.1202412.
[24] S. Swaminathan, R. Tessier, D. Goeckel, W. Burleson, A dynamically reconfigurable adaptive viterbi decoder, in: Proc. Of International Symposium on Field Programmable Gate Arrays, 2002, pp. 227–236, http://dx.doi.org/10.1145/503048.503081.
[25] J. Nargis, D. Vaithiyanathan, R. Seshasayanan, Design of high speed low power Viterbi decoder for TCM system, in: Proc. of the International Conference on Information Communication and Embedded Systems – ICICES 2013, February 2013, pp. 185–190, http://dx.doi.org/10.1109/ICICES.2013.6508239.
[26] Xilinx Inc., San Jose, CA, Virtex-6 FPGA data sheet, January 19, 2012, <http://www.xilinx.com/support/documentation/data_sheets/ds150.pdf>.
[27] Cadence, Encounter user guide, Version 6.2.4, March 2008.