Power-Efficient Viterbi Decoder Architecture and Field Programmeble Gate Arrays Fpga Implementation

Burcu Özbay, Serap Çekli

Department of Computer Engineering, Maltepe University School of Engineering, İstanbul, Turkey

Cite this article as: B. Özbay, S. Çekli. "Power-Efficient Viterbi Decoder Architecture and Field Programmeble Gate Arrays Fpga Implementation". *Electrica*, vol. 18, no. 1, pp. 52-59, 2018.

ABSTRACT

A Viterbi decoder system comprises a convolutional encoder and Viterbi decoder. In general, the code words generated from the input series of convolutional encoder arrive at the decoder through a noisy channel; however, the channel noise can cause corruption of code words. The Viterbi decoder extracts the original input message from the corrupted data using the Viterbi algorithm based on the maximum likelihood principle. A Viterbi decoder mainly comprises four essential units: a branch metrics unit, add-compare-select unit, path metrics unit, and survivor-path memory unit. Related complex calculations are repeated in these units at each clock cycle. In this study, a power- and area-efficient Viterbi decoder architecture that also reduces the computational complexity is proposed. Initially, a hard-decision Viterbi decoder system architecture design for Very Large Scale Integration (VLSI) realization was fulfilled without any further improvement to compare the performance of fundamental and improved designs with respect to power consumption. The initial design constitutes an essential base for the improved power- and area-efficient Viterbi decoder architecture. The improvements were made to achieve the less complex and power-efficient architectural system design. The performance of the proposed architecture was tested by a fieldprogrammable gate array (FPGA) platform, and the results have been reported. The architectural design is described using the Verilog hardware description language for comparing the related tests and performance of FPGA platform.

Keywords: Viterbi decoder architecture, FPGA implementation, forward error correction

Introduction

The transmission channel noise is the most influential and important problem on the impeccable data transmission on a communication system. Some error correction coding techniques are used according to the data size, activity field and the priority of the receiver to eliminate the effect of undesired channel noise. The coded data is transmitted to the receiver by decoders after the decoding process. An effective decoder provides intact data transfer after passing through a noisy channel. Viterbi decoders (VD) are the systems based on the aforementioned principle and they are employed frequently in fields such as wireless communication systems, Code Division Multiple Access (CDMA) systems, satellite and space communication systems, Mobile Communication Systems (GSM).

A VD system is comprised of the VD and convolutional encoder which generates forward error correcting code of the convolutional codes. Operation of a convolutional encoder is based on the finite state machine (FSM). Input message bits which provides states of encoder and state transitions could be expressed by using Trellis diagram.

Several approaches have been conducted for the digital system design of the Viterbi decoder, recently. These studies are focused mainly on the increasing the decoding speed [1-3], decreasing the hardware complexity for power efficient hardware design [4-14], changing the constraint length [8-10] and radix4 based digital architectures [15,16].

The designs which aim speeding up VD are carried out by transforming the serial architecture of a conventional Add Compare Select Unit (ACSU) to parallel design with an increase

Address for Correspondence:
Serap Çekli

E-mail:
serapcekli@maltepe.edu.tr

Received: 01.11.2017

Accepted: 28.12.2017

DOI: 10.5152/iujeee.2018.1809

of the number of states of Trellis diagram. Implementation of an architecture which uses that kind of principle has provided that 33% speed up for decoder [1]. Although, this architectural designs make possible the faster decoders, the power consumption is also increased as a result of the increased hardware complexityat the same time.

The studies which trying to provide the power efficiency focuses basicly on the power save by decreasing the hardware complexity. To reduce the hardware complexity, some researches have been done to use an algorithm which decreases the power consumption while tracing back the path for decoding [6,9,10,12]. Moreover, some studies have been put forward to change the architectural design of Survivor Path Memory Unit (SPMU) or ACSU [1,5-7,11,13,14] to decrease the usage of hardware resources.

The most successful approach for the power efficient design of the VD is defining a threshold value for ACSU. In this approach, the calculation load of decoder is reduced by avoiding the paths which have path metrics greater than the defined threshold value. This method is called T-algorithm and some design studies which use this algorithm have been put forward [11,13,14]. Decoder designs which based on this algorithm provides savings from storage areas Path Metrics Unit (PMU)[6]. By using the designs employing this kind of approaches also reduces power consumption by on Xilinx XCV1000 device [7].

In this study, convolutional code which is a forward error correction code is obtained by coding the input message code at first. VD which operates according to hard decision method is designed to turn out this code into non coded form after transmitting through a noisy channel before reaching at receiver. The initial decoder design is changed to power efficient form by some improvements. In this paper, improvement is the use of the T-algorithm. The power efficient decoder designed in this way has achieved 50% power saving in hardware based power consumption on Xilinx XC6SLX16 device.

The rest of the paper is organized as follows. Section 2 presents encoder structure, state diagram and operation of convolutional coder. The background of Viterbi algorithm and Trellis diagram are introduced for the suggested design in Section 2. In Section 3, the designed VD and in Section 4, the designed power efficient VD is proposed.The detailed implementation and comparisons results are given in the Section 5, and this paper concludes in Section 6.

Viterbi Algorithm Background and Convolutional Encoder

A convolutional encoder is based on a finite state machine. The coded sequence is generated from the input information series depending on the current and previous messages. The encoder consists of one or more D flip flops and logic gates as shown in Figure 1. The information on the flip flops changes according to each input message and indicates the current state of the encoder.

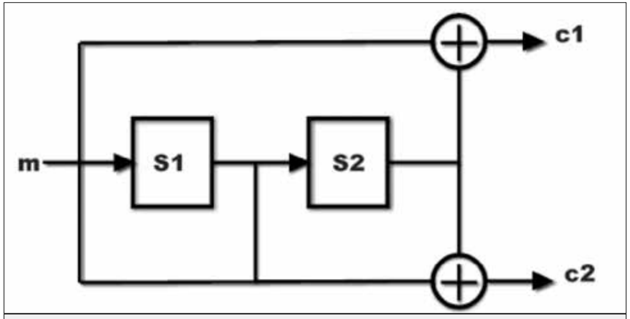

Figure 1. Convolutional encoder

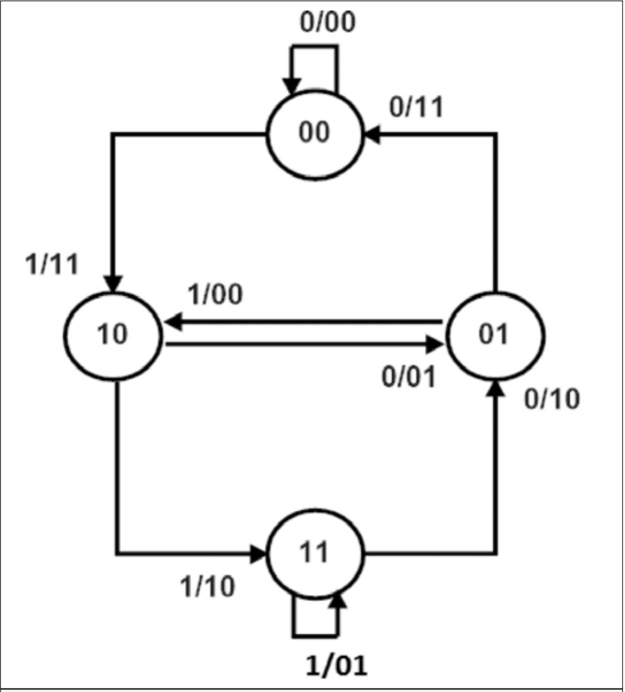

Figure 2. State diagram for the convolutional encoder

A convolutional code is expressed as *Cconv(n,k,K)*, where *k* is the length of the message bit; *K*, the length of the code bit; *n*, the constraint length, the memory of the encoder, and the depth of the code. It shows how many times each input bit is affected on output bit generation. The higher constraint length increases the power of the code but also increases the complexity of the encoder.

The *k*-bit message is collected according to modulo-2 summation, passing through the shift register which *K-1* memory elements, and an -bit output code is generated. The convolutional encoder can be defined by the status table and the trellis diagram. The states are a production of the encoder's shift register. The output code is a function of the message bit to be encoded and the current state.

The state diagram shows the output bits between time instants when an input message is encoded and the time instants when the other input message is encoded. The trellis diagram is a description of the state diagram.

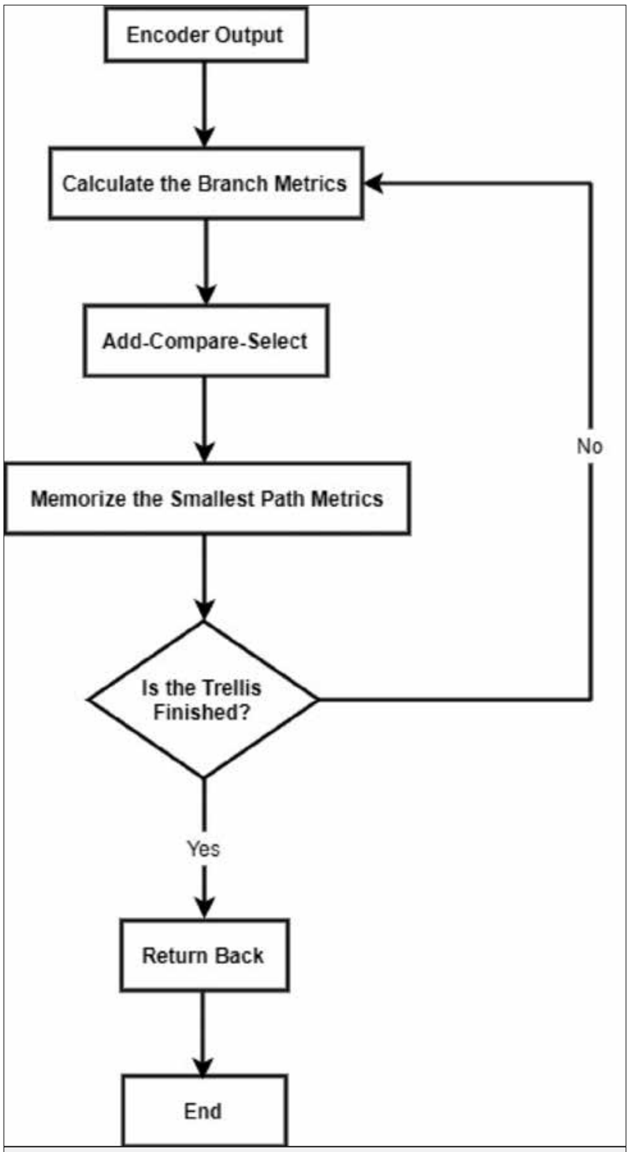

Figure 3. Flowchart of VA

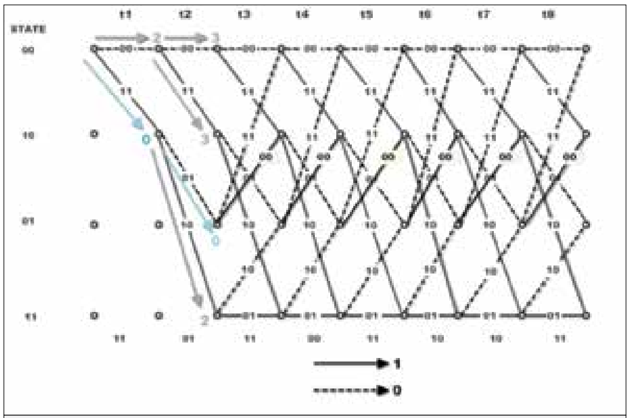

Figure 4. Trellis diagram for the input message sequence "100011" of the encoder

The encoder shown in Figure 1 is expressed as *Cconv*(2,1,3). According to this expression, one bit message is encoded as two bits. Each input message comes to the first memory element of the shift register and the bits which is hold in the memory elements shift to the right one bit. Thus, the encoder pass the next state. The encoder starts the encoding process with zero state. The encoding continues until all the bits in the input message enter the encoder and the encoder returns to zero state again.

The state diagram for the encoder shown in Figure 1 is shown in Figure 2, and the state table for this state diagram is shown in Table 1. This table shows which state encoder pass according to the input message bit and what the exit code is in this moment.

Viterbi Algorithm

Viterbi Algorithm (VA) is based on the Maximum Likelihood (ML) algorithm in solving the convolutional codes. The ML approach finds the most similar way in the trellis diagram for the convolutional encoder. The most similar path is composed of getting together the most similar branches.

Viterbi Algorithm can be defined as the most appropriate algorithm because of its success in reducing the error probability. The algorithm works by following the steps below. The related flowchart of the VA is given in Figure 3.

- The measures paths in the trellis diagram are calculated based on Euclidean Distance (ED) or Hamming Distance (HD).
- In ACSU, the shortest, or most similar path is selected for the code at each encoder output until the data transmission is complete. This path is memorized. These shortest paths are called survivor paths (SP).
- Encoded message against each branch of the selected SP is found by using an appropriate return algorithm.

Trellis diagram

Trellis diagram is generated from the state diagrams. Thanks to the trellis diagram, it is easy to see which output code the encoder produces when it moves from one state to another and which message comes to the encoder's input.

If the constraint length of the encoder is K, the number of states in the trellis diagram is $2^{(K-1)}$ [7].

Each branch in the trellis diagram carries some metrics that represent similarity between the encoded message and the receive message at this moment. According to HD, these values indicate how many bits are different between the recived code and the encoded one. Until the end of the diagram, branch metrics (BM) continue to be collect and thus path metrics (PM) occur. When the trellis diagram is completed, the path which have lowest PM among SPs that will ensure to decode correctly is chosen as the most similar path by VD.

Figure 4 shows the trellis diagram of the message sequence 100011 is encoded by the encoder given in Figure 1. The code sequence of 11 01 11 00 11 10 10 11 is obtained from the output of the encoder with this message sequence.

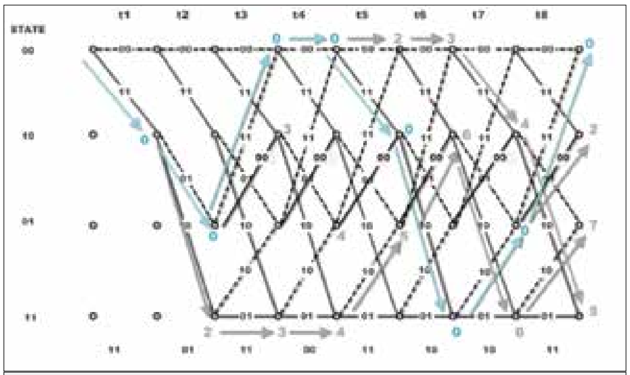

Figure 5. PMs of four possible paths for the input message sequence "100011" of the encoder

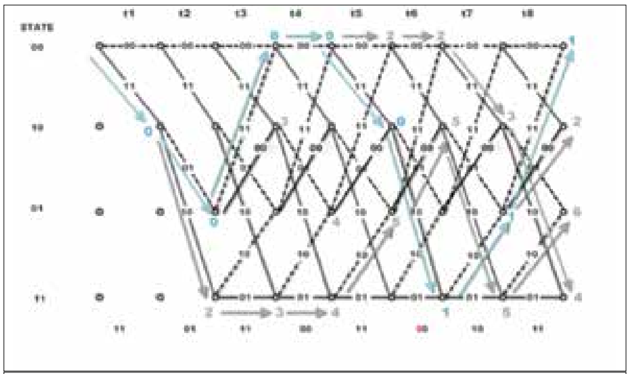

Figure 6. In the case of incorrect receiving, PMs of four possible path for the input message sequence "100011" of the encoder

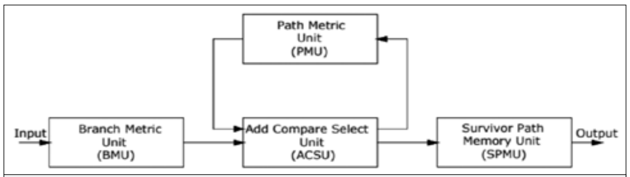
Path Metric Unit (PMU)
Input
Branch Metric Unit (BMU)
Add Compare Select Unit (ACSU)
Survivor Path Memory Unit (SPMU)
Output

Figure 7. Basic units of VD

Trellis steps are continued by collecting the BMs until the encode process is finished. Thus, the PMs of all possible paths are calculated at the end of the trellis diagram. In Figure 5, four paths that can occur as SP when the trellis diagram is completed and their PMs are given. VD, in principle, chooses the route with a low PM. Here, the path with the metric of will be selected and the return algorithm will be applied over that path. Thus, the input code is approached to the highest level.

Assume that the code sequence 11 01 11 00 11 10 10 11 from the encoder output is detected as 11 01 11 00 01 10 10 11 by the decoder due to the noise in the channel. The 9th bit is misunderstood. As shown in Figure 6, because of the this bit error at the 9th bit, BM in the t_5 clock cycle becomes 1, although the smallest total BM in the first four time periods is 0. Due to there are no erroneous bits when the trellis diagram is complete, this path's PM will be 1. Compared to other SPs, the path which have the smallest PM is still this

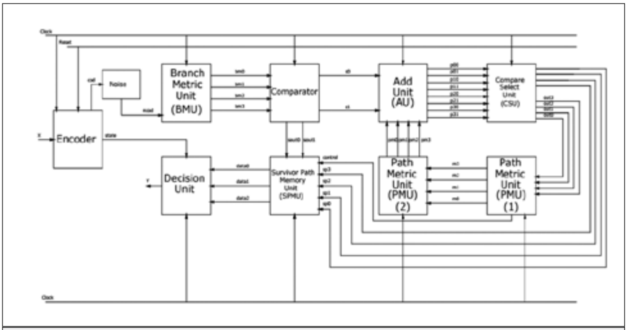
Clock
Reset
Noise
Branch Metric Unit (BMU)
Comparator
Add Unit (AU)
Compare Select Unit (CSU)
Encoder
Decision Unit
Survivor Path Memory Unit (SPMU)
Path Metric Unit (PMU) (2)
Path Metric Unit (PMU) (1)
Clock

Figure 8. Structure of the designed VD

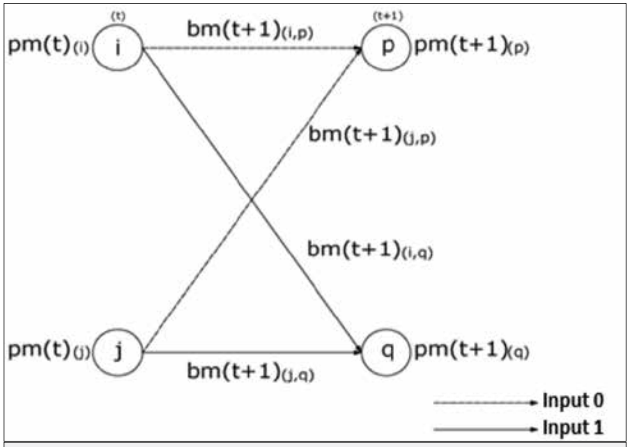

Figure 9. Butterfly structure of the ACSU [17]

path, so the decoder will go back over this path and decode the code 100011. Even though code receive wrong due to the noise in the channel the code is solved correctly thanks to VD.

Proposed Viterbi Decoder Architecture and Implementation

A Viterbi Decoder consists of four basic units as shown in Figure 7. Branch Metric, Add Compare Select, Path Metric, and Survivor Path Memory Unit. Other units different from these units are created by dividing these four basic units, and the design from the design may differ. But on the basis they do what these four units do. The designed VD units and connections between units are shown in Figure 8.

Branch metric unit

The first unit is called Branch Metric Unit. HD or ED can be used to calculate BMs according to the hard or soft decision method. If the decoder is designed according to soft decision method, ED are calculated. If it is designed according to hard decision method, HD are calculated. Although the soft decision decoder provides more information about the input message, the calculation complexity is more.The decoders designed for this study are working using hard decision method because it is aimed at power efficient design due to low complexity. The received code is compared with the encoded values bit by bit for every clock pulse end the difference between them is assigned as BM. The BMs are sent to ACSU to be selected as survivors.

Add compare select unit

ACSU is the unit where current PMs and current BMs are collected, compared and the path with the smallest PMs is selected. BMs are collected by the PMs coming from the previous time interval in this unit. The new PMs generated after this process are compared and the paths with the smallest PM are selected as SPs and sent to the SPMU. In addition, PMs, including this time interval, go to PMU for new calculations.

Figure 9 shows butterfly structure of ACSU. When the decoder is in the state of i and j, it goes to p or q according to the incoming message to the encoder. The PMs in the state of i and j are expressed as pm_t^i and pm_t^j. When BMs which are necessary for transition to p and q state are added to these values, the PMs in the state of p and q are formed and expressed as pm_{t+1}^p and pm_{t+1}^q $bm_{t+1}^{i,p}$, $bm_{t+1}^{j,p}$; are branch values that pass the state i and state j to state p and $bm_{t+1}^{i,q}$, $bm_{t+1}^{j,q}$; are branch values that pass the state i and state j to state q.

The addition process in acs unit is expressed by equations (1), (2), (3), (4) and the comparison process in acs unit is expressed by equations (5) and (6) [17].

$$pm_{t+1}^p = pm_t^i + bm_{t+1}^{i,p} \tag{1}$$

$$pm_{t+1}^p = pm_t^j + bm_{t+1}^{j,p} \tag{2}$$

$$pm_{t+1}^q = pm_t^i + bm_{t+1}^{i,q} \tag{3}$$

$$pm_{t+1}^q = pm_t^j + bm_{t+1}^{j,q} \tag{4}$$

$$\mathrm{pm}_{t+1}^p = \min[(\mathrm{pm}_t^i + \mathrm{bm}_{t+1}^{i,p}),(\mathrm{pm}_t^j + \mathrm{bm}_{t+1}^{j,p})] \tag{5}$$

$$\mathrm{pm}_{t+1}^q = \min[(\mathrm{pm}_t^i + \mathrm{bm}_{t+1}^{i,q}),(\mathrm{pm}_t^j + \mathrm{bm}_{t+1}^{j,q})] \tag{6}$$

Path metric unit

PMU is the unit where path metrics are calculated at intended time. Path metrics of the SPs are updated in this unit and returned to ACSU to gather new BMs which come from next time interval. The calculations made within the units of the VD shown in Figure 8, including ACSU and PMU are shown in Figure 10.

Survivor path memory unit

SPMU is the unit where the SPs which are traced to decoded are stored. In this work code is decoded by using TB method. The SPMU is needed for this method. The SPs are taken here in memory and the bits encoded per branch are decoded in the reverse order. The TB algorithm is applied on the lowest path of PM.

Proposed Power Efficient Viterbi Decoder Architecture and Implementation

The most common approach for a power-efficient VD is to reduce the computational and hardware complexity. In this study, with the comparator unit which added to the output of the BMU in the first designed decoder, some of the branches with large BM were eliminated and not involved in the calculation. Therefore, the first designed decoder is powerfully active according to a classical VD.

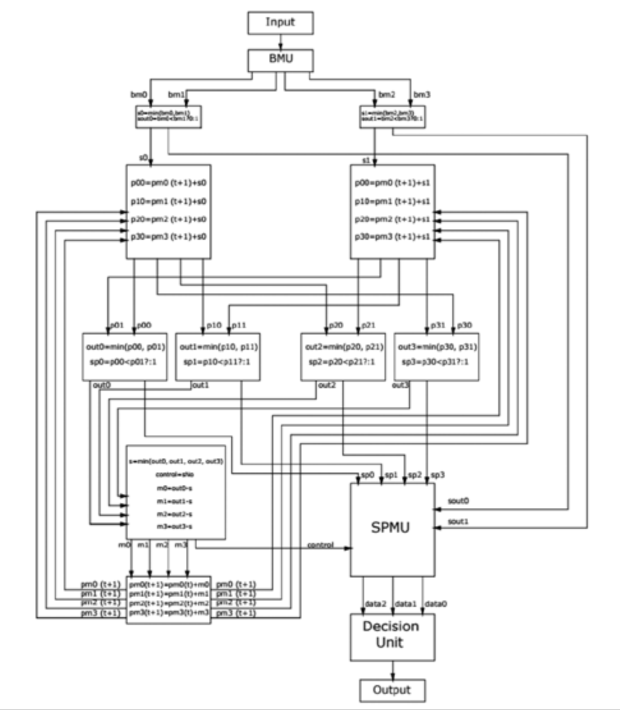

Figure 10. Operations in the ACSU and PMU of the designed VD

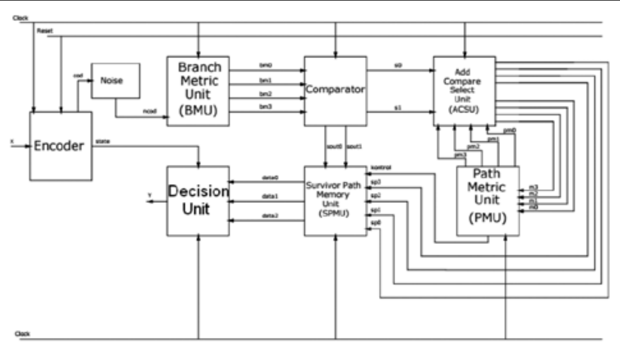

Figure 11. Structure of the designed power efficient VD

The decoder shown in Figure 11 is designed to make the designed decoder more power efficient. A threshold value for PMs has been defined, and the computational complexity of the decoder has been reduced by eliminating the paths that have PMs above this threshold value without comparison. Here, the choice of threshold value is important. Although, a low selected threshold value will reduce the computational complexity at high rate, may lead to the elimination of the path to be decoded. This means that the decoder is far from its most basic aim and it can not solve the code correctly. For this reason, the selected threshold value should be at the optimum level to ensure that the decoder operates most efficiently, without incorrect decoding. The threshold selection should be emprical. It can be attempted to reduce the threshold value until the incorrectly decoded. In this study, according to the input message, the maximum value of the PM is 1. For this reason, threshold is set to 0 which is the best value. Thus, paths with

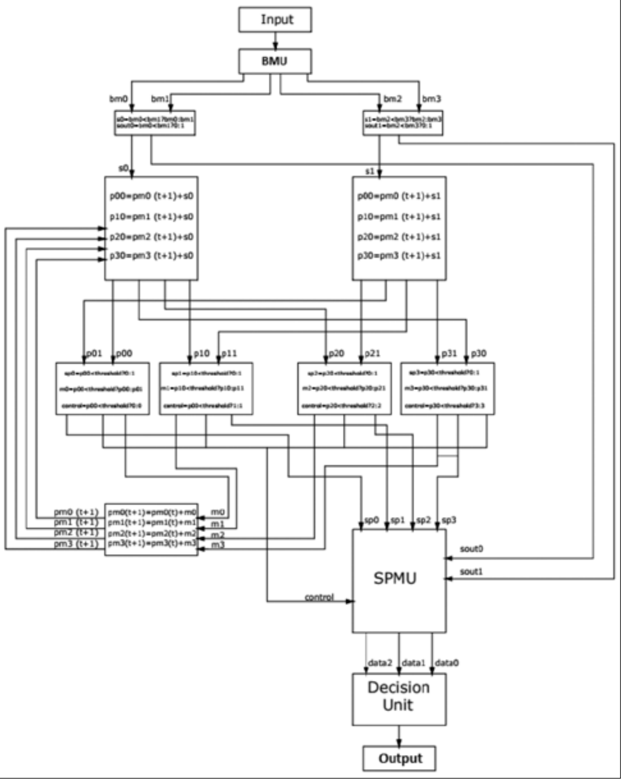

Figure 12. Operations in the ACSU and PMU of the power efficient designed VD

Table 1. State table for the convolutional encoder

Input Message m	State (t) S1 S2	State (t+1) S1 S2	Code c1	Code c2
-	00	00	-	-
0	00	00	0	0
1	00	10	1	1
0	01	00	1	1
1	01	10	0	0
0	10	01	0	1
1	10	11	1	0
0	11	01	1	0
1	11	11	0	1

erroneous decoding have been eliminated and only paths with zero distance to the encoded code are included in the calculation. In situations with large PMs, as the threshold value increases, the number of paths participating in the calculations and computational load of the decoder increases.

As shown in Figure 11, the power efficient VD has the same units as the decoder shown in Figure 8, except that ACSU and PMU are designed as a single unit. The calculations made in all units including ACSU and PMU are shown in Figure 12, and as seen here, the threshold value is defined in ACSU as the 'threshold', starting from here to reduce the calculation load.

Table 2. Viterbi decoder and power efficient Viterbi Decoder hardware utilization

Architectural design	Viterbi decoder	The power efficient viterbi	Hardware reduction
Number of registers	30	6	80%
Number of look up tables	253	110	57%
Number of GCLK (global clock)	1	1	-
Number of input/ output ports	4	4	-

Table 3. Viterbi decoder and power efficient Viterbi Decoder power consumption

Architectural design / Power consumption [W]	Viterbi decoder	The power efficient viterbi	Power reduction
Logic block	0.004	0.002	50%
GCLK	253	110	-
(global clock)	0.005	0.005	-
Input/output port	0.002	0.002	-
Static device	0.020	0.020	-
Total on-chip	0.031	0.028	10%

Results

The number of employed registers and number of look up tables exhibits the hardware complexity and area utilization, evidently for a design which is implemented on the FPGA environment. The more number of register and look up table means the more number of logical operations and logical block utilization. If the aim is to design of a power efficient architecture then the number of registers and look up tables should be reduced due to the hardware complexity. For this purpose, the power efficient VD is designed by defining a threshold value. The PM values are compared with the defined threshold without compared with each other and the paths which are on the threshold are eliminated. In the power efficient design, the number of registers are reduced by and the number of look up tables are reduced by . This reductions are provided by the reduction of the number of PM comparison operations as well as the memory usage for storing the comparison results, and the corresponding comparisons are given in the Table 2. In the power efficient VD design, the less number of logic blocks (LB) are used thus so, the power consumption resulting from the logic blocks is also less.

The power consumption analysis has been performed for the two FPGA implementations which are the VD and power efficient VD design. The power consumption of the VD design is 0.031 *Watt* and the power consumption of the power efficient VD design is 0.028 *Watt*. 0.020 *Watt* power dissipation is constant for the both VD designs and is 0.005 *Watt* also consumed for the clock source. 0.002 *Watt* is consumed for the Input/Output (IO) ports. The power consumption values are given in the Table 3.

When viewed in terms of total power consumption, power saving have correspond a value of 10% in the power-efficient decoder. This can be perceived as small value. But device based power consumption has been 64.5% of the total power consumption of VD and 71.4% of the total power consumption of the power efficient VD. And this static value can not be changed at all. Hence it is more meaningful to examine the power consumption of the logic blocks shown in Table 3. Because in this study the contribution is provided on this part.

The difference between the power consumption characteristics of two designs is resulted from the logic blocks. This result has been expected due to enhancements which are made on the logic block utilization of two designs. The power consumption of the hardware resource employment of the VD design is 0.004 *Watt*, however the power consumption of that of the VD design is 0.002 *Watt*. Therefore, the power consumption arising from the logic blocks has reduced by for the power efficient VD design in this study.

Conclusions

Viterbi decoding is the best technique for decoding the convolutional codes. Convolutional encoder with constraint length 3 and code rate 1/2 and the decoder decoding this code has been designed and implemented and decoder has been developed as power efficient. The key consideration is to decrease the power dissipation. In accordance with this purpose this paper has presented T-algorithm for power efficient management in VD. The ACSU consumes most of the power. The hardware complexity of the decoder is reduced by defining an appropriate selected threshold value in the ACSU and a power efficient VD which has the same ability to decode successfully as the first designed decoder in this study is obtained. Power consumption on hardware has been reduced to half. The different modules are designed using Verilog HDL (Hardware Description Language) and maped to Xilinx XC6SLX16 using Xilinx Integrated Software Environment (ISE).

References

1. I. Lee and J.L. Sonntag, "A new architecture for the fast Viterbi algorithm," IEEE Global Telecommunications Conference, San Francisco, vol. 3, pp. 1664-1668, Nov., 2000.
2. M. Benaissa, and Y. Zhu, "A Novel High-Speed Configurable Viterbi Decoder for Broadband Access," *EURASIP Journal on Applied Signal Processing,* pp. 1317-1327, 2003. [CrossRef]
3. Y. Zhu and M. Benaissa, "Reconfigurable Viterbi decoding using a new ACS pipelining technique," Proceedings of the 2003 IEEE international conference on Application-Specific Systems, Architectures, and Processors (ASAP2003), The Hague, The Netherlands, pp. 360-368., 24-26 June 2003. [CrossRef]
4. K. Page, P. M. Chau, "Improved Architectures for the Add-Compare-Select Operation in Long Constraint Length Viterbi Decoding," *IEEE J. Solid-State Circuits,* vol. 33, pp. 151-155, no. 1, January 1998.
5. I. Kang and A. Willson, "Low-Power Viterbi Decoder for CDMA Mobile Terminals," *IEEE J. of Solid-State Circ.,* vol. 33, no. 3, pp. 473-482, Mar. 1998. [CrossRef]
6. Y. Gang, T. Arslan, A. T. Erdogan, "An Efficient Pre-Traceback Approach for Viterbi Decoding in Wireless Communicationl" ISCAS 2005, *IEEE International Symposium on Circuits and Systems,* vol. 6, pp. 5441- 5444, May, 2005.
7. R. Tessier, S. Swaminathan, R. Ramaswamy, D. Goeckel, W. Burleson, "A Reconfigurable, Power-Efficient Adaptive Viterbi Decoder," *IEEE Transactions On Very Large Scale Integration (VLSI) Systems,* vol. 13, no. 4, pp. 484-488, April, 2005.
8. A. Dinhand and H. Xiao, "A Hardware-Efficient Technique to Implement a Trellis Code Modulation Decoder," *IEEE Transactions On Very Large Scale Integration (VLSI) Systems,* vol. 13, no. 6, pp. 745-750, June, 2005.
9. Y.C. Tang, D. C. Hu, , W. Wei, W. C. Lin, H. Lin, "A Memory-Efficient Architecture for Low Latency Viterbi Decoders," 2009 International Symposium on VLSI Design, Automation and Test, 28-30 April, 2009.
10. K. Cholan, "Design and Implementation of Low Power High Speed Viterbi Decoder," *International Conference on Communication Technology and System Design,* vol.30, pp.61-68, 2011.
11. S.L. Latha and D.L. Kumari, "Low-Power Adaptive Viterbi Decoder for TCM Using T-Algorithm," *International Journal of Scientific and Research Publications,* vol. 3, no. 8, August, 2013.
12. N. Bhatt, M. Shah, B. Asodoriya, "FPGA Implementation Of Power Efficient Low Latency Viterbi Decoder," *Indianternational Journal of Engineering Research & Technology (IJERT)*, vol. 2, no 5, May, 2013.
13. R. I. Thakre, "Design of T-Algorithm Based High-Speed Low-Power Viterbi Decoder for TCM Decoders", *International Journal of Innovative Research in Electronics and Communications (IJIREC),* vol. 1, no. 1, pp. 33-38, April, 2014.
14. V.G. Kumar and A. C. Sudhir, "Implementation of Viterbi Decoder using T-algorithm for TCM Decoders," *International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering,* vol. 3, no 5, May, 2015.
15. P. J. Black and T. H. Meng, "A 140-Mb/s, 32-state radix-4 Viterbi Decoder", *IEEE J. Solid-State Circuits,* vol. 27, pp. 1877-1885, Dec. 1992. [CrossRef]
16. N. Bruels, E. Sicheneder, M. Loew, A. Schackow, J. Gliese, C. Sauer, "A 2.8 Gb/s, 32-State, Radix-4 Viterbi Decoder Add-Compare-Select Unit", *Symposium on VLSI Circuits Digest of Technical Papers,* pp. 170-173, June, 2004. [CrossRef]
17. I.C. Yilmaz, "Design and Simulation of Soft Decision Viterbi Decoder", Department of Electrical and Electronics Engineering, Çukurova University Institute of Natural and Aplied Sciences, MSc. Thesis, 2011.

Burcu Özbay was born in Besni on January 7th, 1990. She graduated from Marmara Private Science High School in 2007. She received her BSc. in Physics Engineering from Hacettepe University in 2012, and her MSc. in Computer Engineering from Maltepe University in 2017. She has been working in Computer Engineering Department of Maltepe University as a research assistant since 2013.

Serap Çekli was born in Germany in 1978. She received her BSc. degree in Electronics Engineering from İstanbul University in 2000, MSc. degree in Electronics and Communication Eng. from İstanbul Technical University in 2003. She has received PhD. degree from İstanbul University, Elect. Electronics Eng. Dept. in 2009. She worked for İstanbul University Engineering Faculty as a research assistant between 2001-2009. She has been working as an assistant professor at Maltepe University, Computer Engineering Department since 2009. Her research interests are digital systems, digital design, computer organization and architecture.