Efficient Architecture for Error Control Coding Using VLSI Implementation

N.K.Shankar [1], T.Senthil [2] & A.Syed Musthaba [3]
[1]Assistant Professor, [2]Assistant Professor, [3]Lecturer
Department of Electronics and Communication Engineering
[1,2]Muthayammal Engineering College, Rasipuram.
[3]Gnanamani College of Technology, Pachal
vlsishankar@gmail.com, t.senthilece@gmail.com , syed.gct@gmail.com

***Abstract*- Viterbi algorithm is widely used as a decoding technique for convolutional codes as well as a bit detection method in storage devices. The design space for VLSI implementation of Viterbi decoders is huge, involving choices of throughput, latency, area, and power. This Paper propose Fast ACS architecture to reduce the area and power of the ACS unit in viterbi decoder. With the proposed structure it is possible to reduce the area and power of the ACS unit by 30% to 40% compare to conventional ACS architecture. The results are based on real designs for which actual synthesis and layouts are obtained using synopsys.**

Index Terms -Viterbi algorithm, VLSI design.

I.INTRODUCTION

The Viterbi decoding algorithm, proposed in 1967 by Viterbi, is a decoding process for convolutional codes in memory-less noise. The algorithm can be applied to a host of problems encountered in the design of communication systems. In addition, the optimization criteria and the design figures keep on changing with the advancement in CMOS technology and design tools.

Different design aspects of the Viterbi decoder have been studied in a number of research papers [1]–[9]. However, most researchers concentrate on one specific component of the design (e.g., path metrics unit or survival memory unit). Somewhat more general studies are presented in [1], [8], and [9]. Still, in authors' view, a systematic and comprehensive analysis summarizing and characterizing as many of the trade-offs and implementation techniques as possible is missing. This contribution presents such a survey, providing designers with clear guidelines and references to find the best solution for every specific case.

II. GENERAL STRUCTURE OF THE VITERBI ALGORITHM

Viterbi decoding algorithm is the most popular method to decode convolutional error correcting codes. In a convolutional encoder, an input bit stream is passed through a shift register. Input bits are combined using the binary single bit addition (XOR) with several outputs of the shift register cells. Resulting output bit streams represent the encoded input bit stream. Generally speaking, every input bit is encoded using output bits, so the coding rate is defined as1/n (or k/n if input bits are used). The constraint length of the code K is defined as the length of the shift register plus one. Finally, generator polynomials G_x define which bits in the input stream have to be added to form the output. An encoder is completely described by n polynomials of degree k or less. A basic flow of viterbi decoder is shown below in Figure 1.

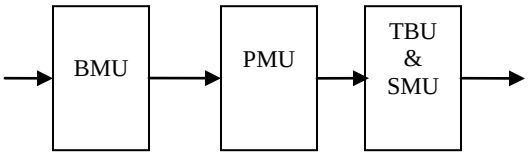

Fig 1 Block Diagram

A. Branch Metric Unit

A branch metric unit's function is to calculate branch metrics, which are average distances between every possible symbol in the code alphabet, and the received symbol.

B. Path Metric Unit

1). Design Space

PMU is a critical block both in terms of area and throughput. The key problem of the PMU design is the recursive nature of the add-compare-select (ACS). Figure 2 shows the ACS Block diagram. In order to increase the throughput or to reduce the area; optimizations can be introduced at algorithmic, word or bit level. To obtain a very high throughput, parallelism at algorithmic level is exploited. By algorithmic transformations, the Viterbi decoding is converted to a purely feed forward computation. This allows independent processing of input blocks. The algorithmic parallel block processing methods intend to achieve unlimited concurrency by independent block decoding of input stream. These techniques result in quite high area figures. But as technological advancements are making the devices shrink, they are getting more attractive. Still, for a specific case, if required throughput can be achieved by utilizing word or bit level optimization techniques, there is no specific need to use algorithmic transformations. Word level optimizations work on folding (serialization) or

unfolding (parallelization) the ACS recursion loop. In the folding technique, the same ACS is shared among a certain set of states. This technique trades off throughput for area. This is an area efficient approach for low throughput decoders, though in case of folding, routing of the PMs becomes quite complex. With unfolding, two or more trellis stages are processed in a single recursion (this is called look ahead).

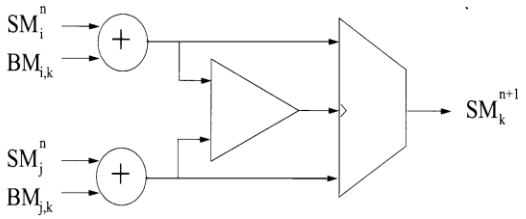

Fig 2 ACS unit for the Viterbi decoder

2). Path Metric Precision

The register temporarily storing path metrics should be wide enough to avoid overflow errors in the PMU operations. Modulo arithmetic is usually used for this purpose. PM bit width is determined as follows:
$PM_{BW} = [\log_2 B + \log_2(2x2x(K-1))]$

C. Trace Back Unit

Trace Back Unit restores a maximum-likelihood path from the decisions made by PMU. Since it does it in inverse direction, a viterbi decoder comprises a FILO (first-in-last-out) buffer to reconstruct a correct order.

D. Survivor Memory Unit

The survivor path storage block is necessary only for the trace back approach. The block records the survivor path of each state selected by the ACS module. It requires one bit of memory per state per stage to indicate whether the survivor path is the upper one or the lower one. The Combined TBU (TRACE BACK UNIT) and SMU (SURVIVOR MEMORY UNIT) generates the decoded outputs.

III. NEW STRUCTURE: DOUBLE STATE: FAST ACS METHOD

The proposed FAST ACS method uses clock gating technique for low power and also it replaces the existing ACS in path metric unit by placing new ACS architecture. The new method accepts the branch metric values with equal weights which are a single state architecture for the butterfly processing unit of the path metric unit it can also be calculated for Double state. The Fast ACS structure is shown in Figure 3.

For simplicity, we assume a binary input case (m=2). This result can be easily generalized to a multiple-input level case. Also, we continue explaining a new structure in the MLSD case. Applying the same structure to the ML convolutional decoder is straightforward.

First, note that the channel response polynomial H(D) of N order could be written as $H(D) = h_0+h_1D+\ldots+h_ND^N+0.D^{N+1.}$ The numbers next to the states represent the input sequence. Also, the numbers on the arrow line show the ideal channel output associated with the transition.

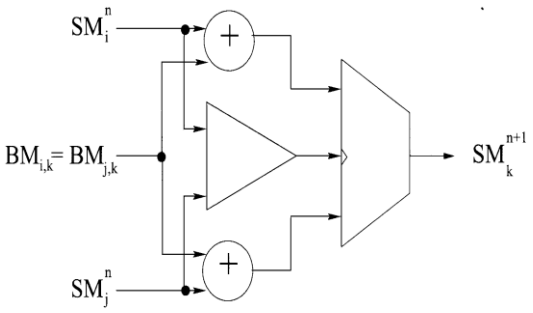

Fig 3 A Fast ACS unit

For a channel H (D) which has a zero coefficient for the last coefficient, the BMs for two transitions which have the same ending state are the same, because the two starting states are different in only the oldest bit position. In this case, the Viterbi processor 2^{N+1} has H (D) states, even though is actually a polynomial of order N (thus the term "double state").

By having the double state in a trellis, the BMs ending in one state are all the same. This means that when choosing the minimum of two possible SMs $SM_i^n + BM_{i,k}$ and $SM_j^n + BM_{j,k}$, we can select the less of two previous SMs SM_i^n and SM_j^n without waiting for an addition of the BMs. (In this case, $BM_{i,k} = BM_{j,k}$) Equivalently, we perform the following recursion.

$$SM_k^{n+1} = \min (SM^n_i) + BM_k^n$$

For example, in the current state 00 of Fig. 3, two incoming paths from the previous states 00 and 10 have the same BM 0. This applies to all the other states, since in the double state, the oldest input to the Viterbi processor makes no contribution on computing the BM for each state transitions.

Therefore, in the double-state structure, the "Add" operation which computes the SM can be carried out at the same time as the "Compare" operation. This new structure is shown in Fig. ---. As clearly shown in this diagram, two BMs $BM_{i,k}$ and $BM_{j,k}$ are the same. A combined ACS unit is shown below in Figure 4.

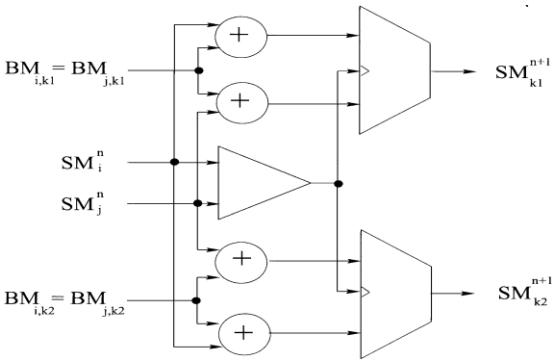

Fig 4 A combined ACS units.

IV.SIMULATION RESULTS

Our proposed Viterbi decoder is coded in VHDL and synthesized with Synopsys tool using the CMOS 120µm technology library. A 2.W supply voltage is assumed. In order to measure the effect of each of the applied techniques on power consumption of Viterbi decoder, each of the mentioned techniques is implemented in a separate circuit. Each circuit is simulated with a representative set of input vectors. Transitions in different nodes are recorded in an output file which is read by Synopsys power estimation tool. This information besides the power attributes of each gate which is extracted from Synopsys library is used to estimate power consumption of the circuit.

The Area and Power comparison of the ACS unit and Fast ACS are given in the Table 1.

Table I Power and Area comparison of ACS units

Unit	**Area (µm^2)**		**Power(µw)**	
	Cell	Total	Total cell leakage	Total dynamic
ACS	205	235.23	1.006	462.59
FAST ACS	98	104.37	0.427	246.62

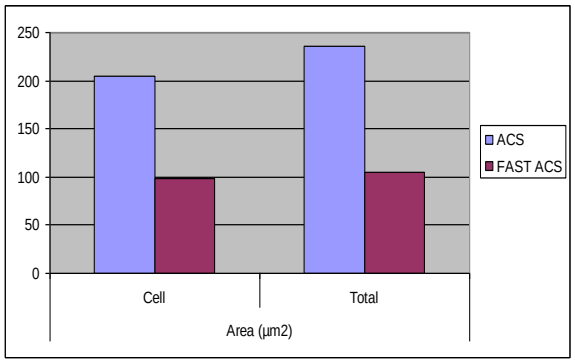

Fig 5 Power comparison of ACS units

The Area and Power comparison of the Different Decoder having ACS and Fast ACS unit are given in the Table 2.

Table II Power and Area comparison of Decoder unit

Unit	**Area (µm^2)**		**Power(µw)**	
	Cell	Total	Total cell leakage	Total dynamic
ACS	1586.50	1946.73	8.085	663.12
FAST ACS	1500.75	1864.68	7.677	593.83

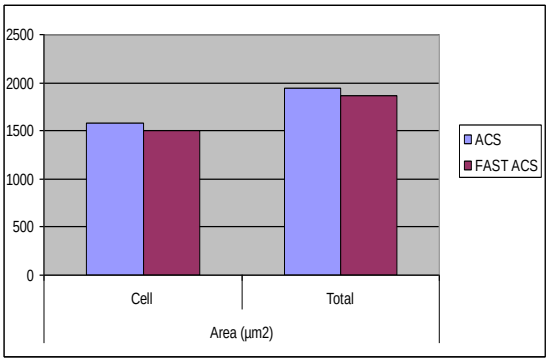

Fig 6 Power comparison of Decoder unit

V.CONCLUSIONS

In this paper, a comprehensive analysis of the Viterbi decoder design space is presented. Table I and II summarizes the importance of the different subunits of the decoder depending on the optimization criteria. The most significant contributions of the paper can be summarized as quantitative comparison of different ACS architectures; in particular, the Fast ACS unit and the conventional Radix 2 ACS are been Compared. For a better overview of the material, Table I and II summarizes the charts and tables related to different design aspects to choose the Fast ACS Viterbi decoder design over the conventional Radix 2 ACS.

REFERENCES

[1] Irfan Habib, Özgün Paker, *Member, IEEE*, and Sergei Sawitzki, *Member, IEEE* "Design Space Exploration of Hard-Decision Viterbi Decoding: Algorithm and VLSI Implementation" IEEE transactions on Very Large Scale Integration (VLSI) systems, vol. 18, no. 5, May 2010.

[2] inkyu lee senior member IEEE, and Jeff L. Sonntang"A New Architecture for the Fast Viterbi Algorithm"IEEE transactions on Communications vol 51.,no 10 october 2003.

[3] F. Angarita, M. J. Canet, T. Sansaloni AND J. Valls" Architectures for the Implementation of a OFDM-WLAN Viterbi Decoder" Springer ScienceJournal of Signal Processing Systems 52, 35–44, 2008.

[4] G. Fettweis, H. Meyr, "Cascaded feedforward architecture for parallel Viterbi decoding,"*IEEE ISCAS*, 978-81, *subm. Kluwer J. VLSI Sig. Proc.* 1990.