Low-Power Adaptive Viterbi Decoder for TCM Using T-Algorithm

MUCHHUMARRI SANTHI LATHA*, Smt. D.LALITHA KUMARI**

* Department Of ECE, JNTU ANANTAPUR
** Department Of ECE, JNTU ANANTAPUR

ABSTRACT: Viterbi Decoder (VD) employed in digital wireless communication plays a dominant role in the overall power consumption of trellis coded modulation (TCM) decoder. Power reduction in VD could be achieved by reducing the number of states. A pre-computation architecture with T-algorithm was implemented for this purpose, and when we compare this result with full Trellis VD, this approach significantly reduces power consumption without degrading decoding speed much .

Low power design of VD for TCM systems with reliable delay is presented in this paper. This work focuses on the realization of convolutional encoder and adaptive Viterbi decoder (AVD) with a constraint length (K) of 9 and a code rate (k/n) of ½ using field programmable gate array (FPGA) technology. The performance of the implemented AVD is analyzed by using ISE 10.1 and Modelsim simulations.

Index Terms: Viterbi decoder ,convolutional encoder, FPGA.

I. INTRODUCTION

The reliability and efficiency of data transmission is the most concerning issue for communication channels in today's digital communications, Error correction technique plays a very important role in communication systems.

Convolutional encoding with viterbi decoding can be used as a Forward error correction technique and this approach provides good performance with low cost and is particularly suited to a channel in which the transmitted signal is corrupted mainly by additive white gaussian noise (AWGN).

Trellis coded modulation (TCM) employs a high-rate convolutional code as they are used in bandwidth-efficient systems. This leads to a high complexity of the Viterbi decoder (VD) for the TCM decoder, even if the constraint length of the convolutional encoder is moderate. Due to the large number of transitions in the trellis diagram power consumption is more in VD.

In order to reduce the power consumption as well as the computational complexity, low-power schemes should be exploited for the VD in a TCM decoder. T-algorithm was shown to be very efficient in reducing the power consumption. However, searching for the optimal PM in the feedback loop still reduces the decoding speed. To overcome this drawback, two variations of the T-algorithm have been proposed: the relaxed adaptive VD , which suggests using an estimated optimal PM, instead of finding the real one each cycle and the limited-search parallel state VD based on scarce state transition (SST). Because of some drawbacks in both of them, we proposed an add-compare-select unit (ACSU) architecture based on pre-computation incorporating T-algorithm for VDs. This can efficiently improves VDs clock speed

II. CONVOLUTIONAL ENCODING WITH VITERBI DECODING

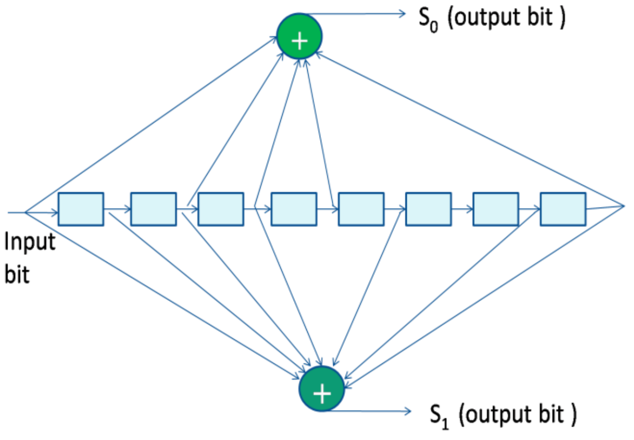

Figure 1: Convolutional encoder with K= 9 and k/n =½

A convolutional code is a type of error-correcting code which contains memory and the *n* encoder outputs at any given time unit depend not only on the *k* inputs at that time unit but also on m previous input blocks. Convolutional codes are usually characterized by two parameters code rate (k/n) and constraint length (K) and the patterns of *n* modulo-2 adders. The shift register has a constraint length (K) of 9, equal to the number of stages in the register The encoder has *n* generator polynomials, one for each adder. An input bit is fed into the leftmost register. Using the generator polynomials and the existing values in the remaining registers, the encoder outputs *n* bits. The code rate (k/n) is expressed as a ratio of the number of bits into the Convolutional encoder *k* to the number of channel symbols output by the Convolutional encoder *n* in a given encoder cycle.
Convolutional encoder with constraint length 9 and code rate ½ is shown in the fig 1. For this encoder we perform the decoding process

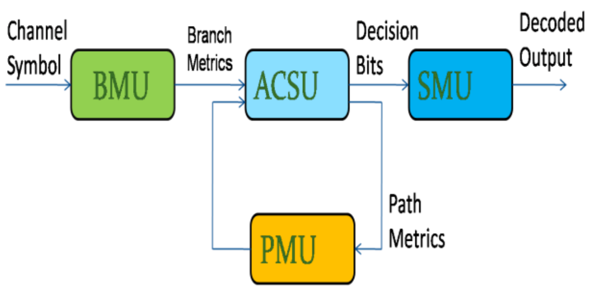

Figure2 : General VD functional diagram.

Viterbi decoder:

Viterbi decoder functional block diagram is shown in Fig.2. Branch metrics (BMs) are calculated in the BM unit (BMU) from the received symbols..Then, BMs are fed into the add-compare-select unit (ACSU) that recursively computes the PMs and outputs decision bits for each possible state transition. After that, the decision bits are stored in and retrieved from the SMU in order to decode the source bits along the final survivor path. The PMs of the current iteration are stored in the PM unit (PMU). ACSU
implementation is critical because the feedback loop makes it the bottleneck for high speed applications. Furthermore, as K value increases, the power consumption and computation complexity increase exponentially. In the T-algorithm, a threshold T is set and the difference between each PM and the optimal one is calculated. So T-algorithm requires extra computation in the ACSU loop for calculating the optimal PM (minimum value of all PMs)and puncturing states.

III. PRE-COMPUTATION ALGORITHM

VD for a convolutional code with a constraint length K contains 2^{k-1} states and consider each state receives p candidate paths. First, we expand PMs at the current time slot n (PMs(n)) as a function of PMs(n-1) to form a look-ahead computation of the optimal PM - $PM_{opt}(n)$. The Branch metric can be calculated by two types: Hamming distance and Euclidean distance If the branch metrics are calculated based on the Euclidean distance, $PM_{opt}(n)$ is the minimum value of PMs(n) obtained as
$PM_{opt}(n)$

$$= \min \{ PM_0(n) , PM_1(n) ,\ldots\ldots\ldots PM_{2^{k-1}}(n)$$

$$= \min \{ \min [PM_{0,0} (n-1)+ BM_{0,0} (n),$$

$$PM_{0,1} (n-1)+ BM_{0,1} (n) ,\ldots\ldots\ldots$$

$$PM_{0,P}(n-1)+ BM_{0,P} (n)] ,$$

$$\min [PM_{1,0} (n-1)+ BM_{1,0} (n),$$

$$PM_{1,1} (n-1)+ BM_{1,1} (n) ,\ldots\ldots\ldots$$

$$PM_{1,P}(n-1)+ BM_{1,P} (n)] ,$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$\min [PM_{2^{k-1},0} (n-1)+ BM_{2^{k-1},0} (n),$$

$$PM_{2^{k-1},1} (n-1)+ BM_{2^{k-1},1} (n) \ldots.$$

$$PM_{2^{k-1},P}(n-1)+ BM_{2^{k-1},P} (n)] \} \quad (1)$$

For a VD usually the trellis butterflies have a symmetric structure. To reduce the computational overhead caused by look-ahead computation we group the states into several clusters. The states can be grouped into m clusters, where all the clusters have the same number of states and all the states in the same cluster will be extended by the same BMs, The min(BMs) for each cluster can be easily obtained from the BMU or TMU (In a TCM decoder, BMU is replaced by transition metrics unit (TMU), which is more complex than the BMU) and the min(PMs) at time n-1 in each cluster can be precalculated at the same time when the ACSU is updating the new PMs for time n.

The precomputation scheme can be extended to q steps, where q<n (q being any positive integer) Hence, $PM_{opt}(n)$ can be calculated directly from PMs(n-q)in q cycles.

The above algorithm (1) when implemented in form of clusters can be rewritten as

$$PM_{opt}(n) = \min \{ \min(PMs(n-1) \text{ in cluster } 1)$$

$$+ \min(BMs(n) \text{ for cluster } 1),$$

$$\min(PMs(n-1) \text{ in cluster } 2)$$

$$+ \min(BMs(n) \text{ for cluster } 2),$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$\min(PMs(n-1) \text{ in cluster } m)$$

$$+ \min(BMs(n) \text{ for cluster } m) \} \quad (2)$$

Choosing Precomputation Steps :

In a TCM system, the convolutional code usually has a coding rate of R/R+1 and the logic delay of the ACSU is $T_{ACSU}= T_{adder} +T_{p\text{-}in\text{-}comp}$ .If T-algorithm is employed in the VD, the iteration bound is slightly longer than T_{ACSU} because there will be another two-input comparator in the loop to compare the new PMs with a threshold value obtained from the optimal PM and a preset T and is given by

$$T_{bound} = T_{adder} +T_{p\text{-}in\text{-}comp}+ T_{2\text{-}in\text{-}comp} , \quad (3)$$

where T_{adder} is the logic delay of the adder to compute PMs of each candidate path that reaches the same state and $T_{p\text{-}in\text{-}comp}$ is the logic delay of a p-input comparator (where $p=2^{R}$) to determine the survivor path for each state.

q-step precomputation can be pipelined into q stages, where the logic delay of each stage is continuously reduced as q increases. As a result, the decoding speed of the low-power VD is greatly improved. However, after reaching a certain number of steps, q_b further precomputation would not result in additional benefits because of the inherent iteration bound of the ACSU loop.

We limit the comparison to be among only p or 2p metrics, to achieve the iteration bound expressed in (3), for the precomputation in each pipelining stage and assume that each stage reduces the number of the metrics to 1/p (or 2^{-R}) of its input metrics. The smallest number of precomputation steps (q_b) meeting the theoretical iteration bound should satisfy

$$[(2] \uparrow R)^\wedge q_b \geq 2^{k-1}. \qquad (4)$$

Therefore $q_b \geq \frac{k-1}{R}$ and we express this as $q_b = \left[\frac{k-1}{R}\right]$

where [.] denotes ceiling function.

Computational overhead is an important factor that should be carefully evaluated. If there are m remaining metrics after comparison in a stage, the computational overhead from this stage is at least m addition operations. For a code with a constraint length k and q precomputation steps,the number of metrics will reduce at a ratio of $2^{\frac{k-1}{q}}$ and the overall computational overhead is $N_{overhead}$. (5)
The estimated computational overhead increases exponentially to q. In a real design, the overhead increases even faster. Therefore, a small number of precomputational steps is preferred even though the iteration bound may not be fully satisfied. One- or two-step precomputation is a good choice in most cases. For TCM systems, where high-rate convolutional codes are always employed, two steps of precomputation could achieve the iteration bound and also it reduces the computational overhead .

IV. LOW- POWER VITERBI DECODER DESIGN

For 4-D 8PSK TCM system with code rate ½, since the precomputation algorithm always finds the accurate optimal PM, its BER performance is almost same as that of the conventional T-algorithm .

ACSU DESIGN :

Convolutional encoder with *Rate ½ and length 9* is shown in fig(1). For convenience of discussion, we define the left-most register in Fig. 1 as the most-significant-bit (MSB) and the right-most register as the least-significant-bit (LSB).The 256 states and PMs are labeled from 0 to 255. The two-step pre-computation in the ACSU feedback loop is expressed as

$PM_{opt}(n)$ =

Min **[** min {
min (cluster 0 (n-2)+ min (BMG0(n-1)),
min (cluster 1 (n-2)+ min (BMG1(n-1)),
min (cluster 2 (n-2)+ min (BMG3 (n-1)),
min (cluster 3 (n-2)+ min (BMG2(n-1)) }
+min(even BMs(n)),
min {
min (cluster 0 (n-2)+ min (BMG1(n-1)),
min (cluster 1 (n-2)+ min (BMG0(n-1)),
min (cluster 2 (n-2)+ min (BMG2 (n-1)),
min (cluster 3 (n-2)+ min (BMG3(n-1)) }
+min(odd BMs(n)) **]**
Where

Cluster 0 = {PM_m | $0 \le m \le 255$, m mod 4 =0 };

Cluster 1= { PM_m | $0 \le m \le 255$, m mod 4 =2};

Cluster 2= { PM_m | $0 \le m \le 255$, m mod 4 =1 };

Cluster 3= { PM_m | $0 \le m \le 255$, m mod 4 =3 };

BMG0= { BM_m | $0 \le m \le 15$, m mod 4 =0 };

BMG1= { BM_m | $0 \le m \le 15$, m mod 4 =2};

BMG2= { BM_m | $0 \le m \le 15$, m mod 4 =1 };

BMG3= { BM_m | $0 \le m \le 15$, m mod 4 =3};

The functional block diagram of the VD with two-step precomputation T-algorithm is shown in Fig. 3. The minimum value of each BM group (BMG) can be calculated in BMU or TMU and then passed to the "Threshold Generator" unit (TGU) to calculate (PM_{opt}+T). (PM_{opt}+T)and the new PMs are then compared in the "Purge Unit"(PU). The architecture of the TGU is shown in Fig. 4,

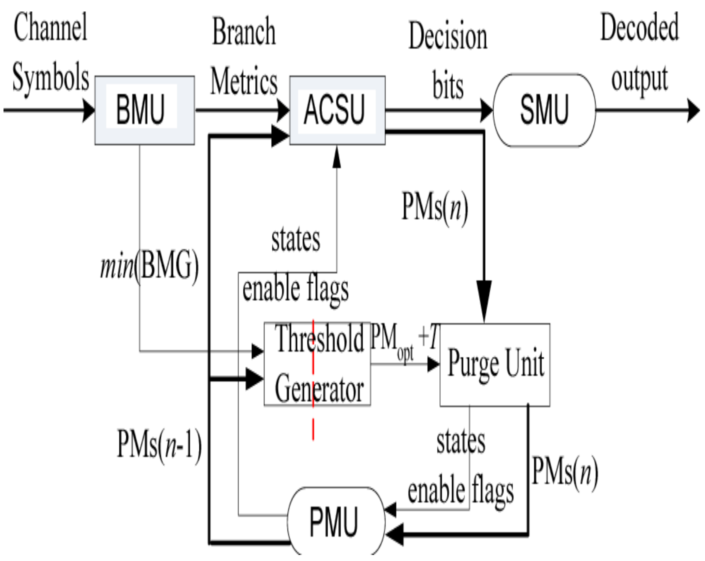

Figure 3 : 2-step pre-computation T –algorithm for VD

SMU Design:

There are two different types of SMU in the literature: trace back (TB) schemes and register exchange (RE) schemes. In the regular VD without any low-power schemes, SMU always outputs the decoded data from a fixed state if RE scheme is used, or traces back the survivor path from the fixed state if TB scheme is used. For VD incorporated with T-algorithm, no state is guaranteed to be active at all clock cycles. Thus it is impossible to appoint a fixed state for either outputting the decoded bit (RE scheme) or starting the trace-back process (TB scheme). In the conventional implementation of T-algorithm , the decoder can use the optimal state(state with PM_{opt}) which is always enabled, to output or trace back data. As the estimated PM_{opt} is calculated from PMs at the previous time slot, it is difficult to find the index of the optimal state in the process of searching for the PM_{opt} . A 256 –to- 8 priority encoder can be used for this purpose. The output of the priority encoder would be the unpurged state with the lowest index. Assuming the purged states have the flag "0" and other states are assigned the flag "1". Implementation of such direct 256-to-8 is not trivial ,so we employ four 4-to-2 priority encoders for the 256 -to -8 priority encoder. This is shown in Fig. 5. and it is simpler also.

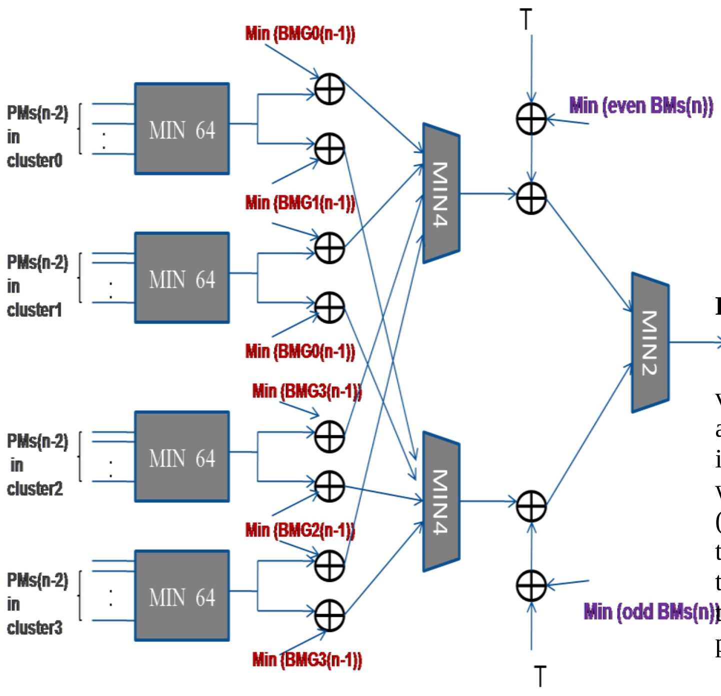

Figure 4: Threshold Generator unit architecture

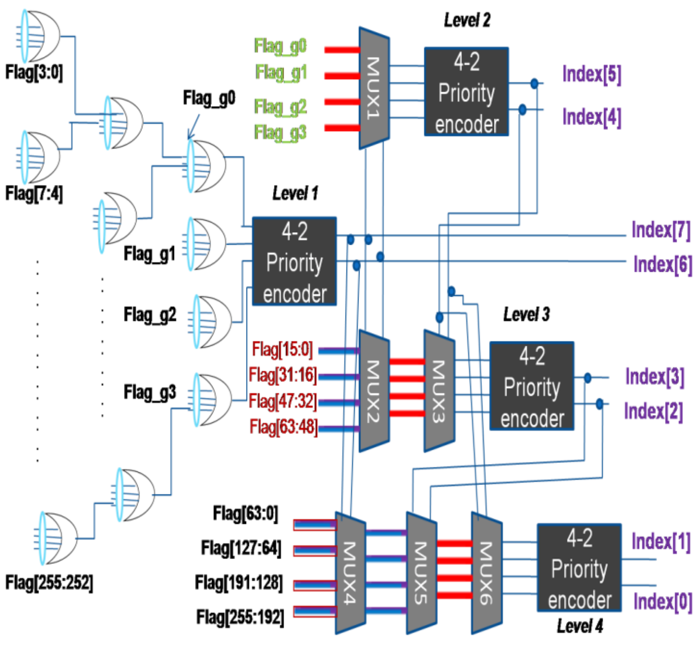

Figure 5: 256 -to -8 priority encoder architecture.

PROCESS ELEMENT TECHNIQUE :

Different protocols use different convolutional code and varied applications have different requirement for throughput, area and power. So design of reusable Viterbi decoder is important, too. In present project, a reusable Viterbi decoder was carried out. This decoder adopted the Process Element (PE) technique, which made it easy to adjust the throughput of the decoder by increasing or decreasing the number of PE. By the method of Same Address Write Back (SAWB), we reduced the number of registers to half in contrast with the method of ping-pong.

This decoder supported punctured convolution code and was data-driven, which means the circuit was able to work under different data rate and avoid those invalid operations. The core parameters, such as the generation words of convolution code, the number of PE, the depth of TBU and maybe the radix of butterfly, are all configurable. By assembling different numbers of PE, we can get the state-serial, part parallel or full parallel structure of Viterbi decoder. And because the PMU is scattered into each PE, this structure is more area efficient.

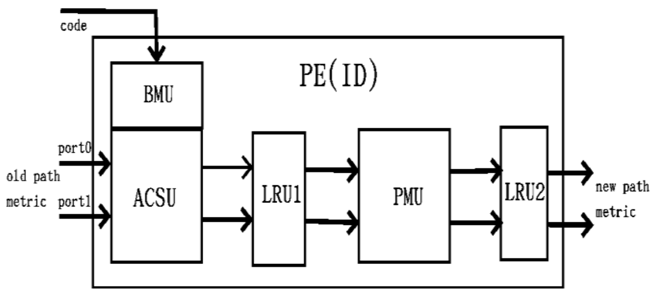

Figure 6: Structure of process element.

5. SYNTHESIS AND SIMULATION RESULT:

Viterbi decoder with rate ½ and K=9 is realized by FPGA device xcv3200e-8fg1156 . The device utilization summary, logic utilization and distribution report is shown in Table I. The precomputation VD reduced the power consumption by nearly 70% with minimum decoding speed.The VD design is simulated by Model Sim and Xilinx ISE 10.1.

Table I : Device Utilization Summary (estimated values)

Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	4106	32448	12%
Number of Slice Flip Flops	2880	64896	4%
Number of 4 input LUTs	7356	64896	11%
Number of bonded IOBs	19	804	2%
Number of BRAMs	16	208	7%
Number of GCLKs	1	4	25%

V. CONCLUSION

The precomputation architecture that incorporates T-algorithm efficiently reduces the power consumption of VDs without reducing the decoding speed appreciably. This algorithm is suitable for TCM systems which always employ high-rate convolutional codes. Both the ACSU and SMU are modified to correctly decode the signal. Compared with the full-trellis VD without a low-power scheme, the precomputation VD could has low power consumption with reliable decoding speed. A reusable Viterbi decoder was carried out by adopting the Process Element technique.

REFERENCES

[1] J. He, H. Liu, and Z. Wang, "A fast ACSU architecture for viterbi decoder using T-algorithm," in *Proc. 43rd IEEE Asilomar Conf. Signals,Syst. Comput.*, Nov. 2009, pp. 231–235.

[2] J. He, Z. Wang, and H. Liu, "An efficient 4-D 8PSK TCM decoder architecture," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 5, pp. 808–817, May 2010.

[3] J. Jin and C.-Y. Tsui, "Low-power limited-search parallel state viterbi decoder implementation based on scarece state transition," *IEEE Trans.Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 11, pp. 1172–1176,Oct. 2007.

[4] J. He, H. Liu, Z. Wang,X.Huang and Kai Zhang ,"High-Speed Low-Power Viterbi Decoder Design for TCM Decoders" in IEEE Trans.on (VLSI) Systems, Vol. 20, No. 4, April 2012.

AUTHORS

MUCHHUMARRI SANTHI LATHA
PURSUING MTECH ,ECE- DECS AT JNTU ANANTAPUR
MAIL ID : :m.santhilatha@gmail.com

UNDER THE GUIDANCE OF
SMT. D.LALITHA KUMARI
ASST. PROFESSOR IN ECE DEPT. JNTUCEA
MAIL ID : lalithad29@gmail.com