A Bit-level ACSU of High Speed Viterbi Decoder

Min Woo Kim and Jun Dong Cho

Abstract—**Viterbi decoder is composed of BMU(Branch metric Unit), ACSU(Add Compare Select Unit), and SMU(Survivor path Memory Unit). For high speed viterbi decoders, ACSU is the main bottleneck due to the compare-select and feedback operation. Thus, many studies have been advanced to solve the problem. For example, M-step look ahead technique and Minimized method are typical high speed algorithms. In this paper, we designed a bit-level ACSU(K=3, R=1/2, 4bit soft decision) based on those algorithms and switched the matrix product order in the backward direction of Minimized method so as to apply Code-Optimized-Array in order to reduce the area complexity. For experimentation, we synthesized our design by using SYNOPSYS Design compiler, with TSMC 0.18 um library, and verified the timing by using CADENCE verilog-XL.**

Index Terms—**Viterbi decoder, ACSU, M-step look ahead, minimized method**

I. INTRODUCTION

Viterbi decoder has been widely used for channel decoding in digital telecommunication. The structure generally consists of BMU (Branch Metric Unit), ACSU (Add-Compare-Select Unit) and SMU (Survivor Path Memory Unit) . The BMU computes branch metrics by hamming distance (or Euclidean distance), and the ACSU computes a summation of branch metric and previous state metric, and then updates state metrics while comparing path metrics [1]. Also, decision vectors are generated. The SMU identifies the original transmitted information on the basis of decision vectors. Here, the ACSU is a main obstacle for high speed because of excessive operation repetitions and feedbacks for updating of state metric. Therefore, lots of researches have been carried out to solve this problem. Among them, M-step Look-Ahead [2] and Minimized-method [3] are well-known high speed algorithm eliminating feedback operations. This paper is organized as follows. First, we will review the look ahead technique and minimized method in Section Ⅱ and next we discuss our enhancement in Section Ⅲ. Finally, we exhibit our design result in Section Ⅳ.

Manuscript received Sep 9, 2006; revised Nov. 9, 2006.
Electronic & Electrical Eng. Dept. Sungkyunkwan University
Chun Chun-Dong, Jang an-Ku, Suwon, 440-746, Korea
E-mail : nakmw@vada.skku.ac.kr

II. ALGORITHM REVIEW

We review M-step look ahead and minimized method algorithm as follows [2, 3].

1. Look Ahead Technique

This algorithm shows that the feedback operation could be dramatically reduced by using matrix product structure. We will show through an example of Trellis diagram which is in case of constraint length K=3, code rate R= 1/2 and hard decision.

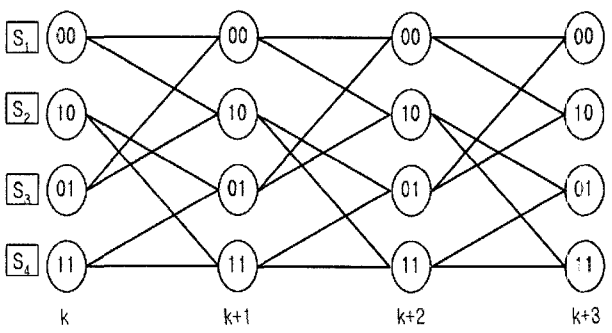

Fig. 1. Trellis diagram (K=3, R=1/2).

$$
\begin{aligned}
S_{1,k+1} &= max(B_{11,k} + S_{1,k}\,;\, B_{13,k} + S_{3,k}) \\
S_{2,k+1} &= max(B_{21,k} + S_{1,k}\,;\, B_{23,k} + S_{3,k}) \\
S_{3,k+1} &= max(B_{32,k} + S_{2,k}\,;\, B_{34,k} + S_{4,k}) \\
S_{4,k+1} &= max(B_{42,k} + S_{2,k}\,;\, B_{44,k} + S_{4,k})
\end{aligned} \tag{1}
$$

The above trellis diagram could be expressed like a Formula 1 and expressed as a linear form through Semiring algebra notation (S_k is state metric and B_k is branch metric.)

$$
\begin{aligned}
S_{1,k+1} &= B_{11,k} \otimes S_{1,k} \oplus B_{13,k} \otimes S_{3,k} \\
S_{2,k+1} &= B_{21,k} \otimes S_{1,k} \oplus B_{23,k} \otimes S_{3,k} \\
S_{3,k+1} &= B_{32,k} \otimes S_{2,k} \oplus B_{34,k} \otimes S_{4,k} \\
S_{4,k+1} &= B_{42,k} \otimes S_{2,k} \oplus B_{44,k} \otimes S_{4,k}
\end{aligned} \tag{2}
$$

And this could be represented by product of matrix-vector as follows. In this case, the matrix-vector operation is not an operation of normal multiplication and addition but an operation of addition and comparison.

$$
S_{k+1} = A_k \otimes S_k
$$

$$
A_k = \begin{bmatrix} B_{11,k} & Q & B_{13,k} & Q \\ B_{21,k} & Q & B_{23,k} & Q \\ Q & B_{32,k} & Q & B_{34,k} \\ Q & B_{42,k} & Q & B_{44,k} \end{bmatrix} \tag{3}
$$

Q implies no change of state transition and column means branch metrics going out from the states of time k and row means branch metric coming into the states of time k+1. And we can derive a new Formula based on Formula (3) as follows.

$$
\begin{aligned}
S_{k+2} &= A_{k+1} \otimes S_{k+1} \\
&= A_{k+1} \otimes (A_k \otimes S_k)
\end{aligned} \tag{4}
$$

The more general form is :

$$
\begin{aligned}
S_{k+M} &= {}_MA_k \otimes S_k \\
({}_MA_k &= A_{k+M-1} \otimes \ldots \otimes A_{k+1} \otimes A_k)
\end{aligned} \tag{5}
$$

In other words, we can get the state metrics if we add state metric of time k with the sum of branch metric from k to k+M. Therefore, we do not have to update state metric in every iteration. It is very easy to see that it is helpful for speed-up. Fig. 2 shows the structure of M-step look ahead. Branch metrics are calculated in A-ACSU and state metrics are updated in M-step ACSU. And we carry out decoding in SMU through trace-back[4]. As shown in Fig. 2, there still exists a feedback operation in M-step ACSU block. This can be resolved by minimized method algorithm.

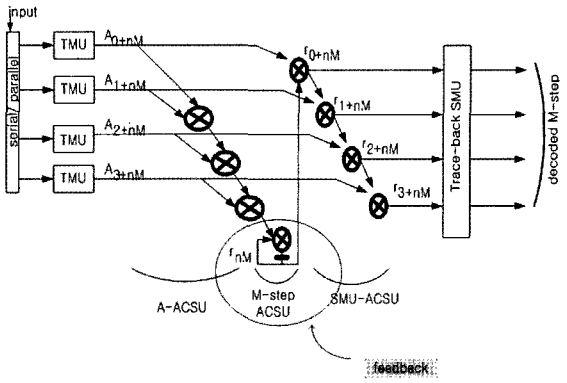

Fig. 2. Structure of the M-step look ahead.

2. Minimized Method

We generally take the survivor depth (decoding depth) as five times of K(encoder constraint length). And we can get a unique path when we decode that depth.(D>5k). So, if we apply M-step look ahead technique (M > D), branch metrics from one state will be connected to all other states.

In this case, when the state metric of decoded state of time k is "a", all other state metrics of time k+M include common value, "a". Those can be normalized by subtracting "a" and this makes no error in decoding process. Because viterbi algorithm is not function of concrete value but function of value's difference.

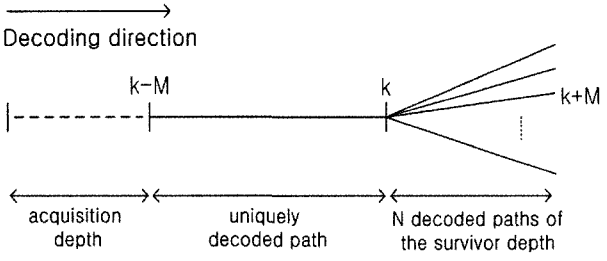

Fig. 3. Scheme of decoded path.

Therefore, we can see that path metrics of time k+M are only sum of branch metrics from k to k+M for the case of M>D. These branch metrics can be implemented by pipeline technique just as it has been seen in M-step look ahead technique. Namely, the summation of branch metric from k to k+M is the same as updating of state metrics in M-step ACSU, and thus the only feedback existing in the M-step ACSU is completely removed. Also all decision values of time k+M have the same results indicating common previous state. So, if we get a decision value from only one state of time k+M , then we know the decoded state in time k. This implies that matrix–vector product operation of ${}_MA_k$ starts from one column of A_k and then computes maximum state of ${}_MA_k$. This can be expressed as followings.

$$\begin{aligned} S_{k+M} &= column_J({}_MA_k) \\ &= A_{k+M-1} \otimes \ldots \otimes A_{k+1} \otimes column_J(A_k) \\ &= {}_MA_k \otimes S_k = {}_MA_k \otimes column_J({}_MA_{k-M}) \end{aligned} \quad (6)$$

As above mentioned, we can get decision value of time k+M in only one state and this can be get through simple operation.

$$\begin{aligned} row_K({}_MA_k) &\otimes column_J({}_MA_{k-M}) \\ row_K({}_MA_k) &= column_K({}_MA_k{}^T) \\ &= A_k{}^T \otimes \ldots \otimes A_{k+M-2}{}^T \otimes column_K(A_{k+M-1}{}^T) \end{aligned} \quad (7)$$

Row of ${}_MA_k$ can be transferred to column through the transpose of matrix. And operation processes of Formula (7) are all the same except for the direction. Row operation carrys out decoding from right to left and column operation carrys out decoding from left to right. Therefore, after summing final state metrics in both directions , the state of maximum state metric is a starting state of decoding which we want to obtain finally. And while starting decoding again from the state, we can get both decision vectors and decoded bits by using trace-back algorithm (Fig. 4)

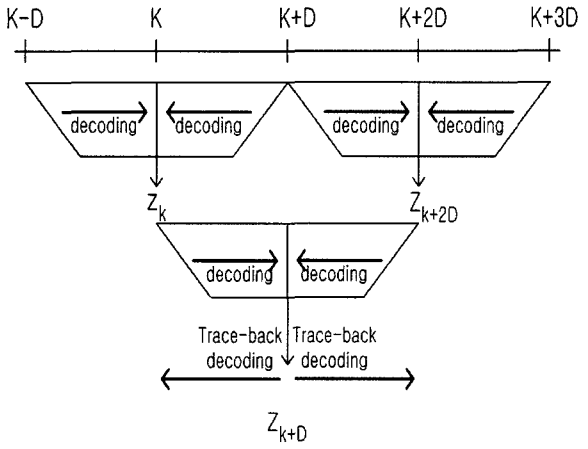

Fig. 4. Scheme of minimized method.

III. MODIFICATION

We designed a core block of minimized method by bit-level PE (Processing Element) and code optimized array suggested in M-step look ahead technique.

1. Modified Processing Element

We removed the redundant bit (when we express cs number 1, there are two ways as 01, 10) from carry save number (0,1,2) by code converter and used another comparison technique[5]. It reduced area of PE. And by using code optimized array, timing of output was changed.

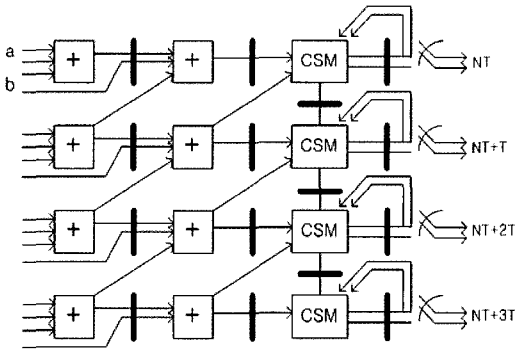

Fig.5. Proposed processing element.

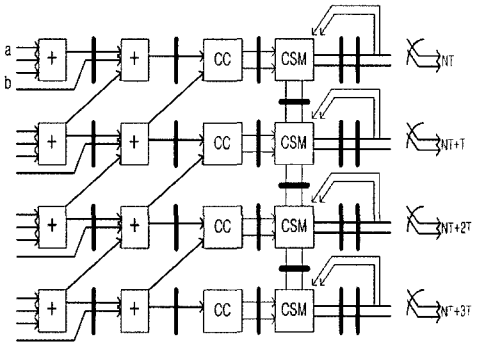

Fig. 6. Modified processing element(CC:code converter,4-bit level).

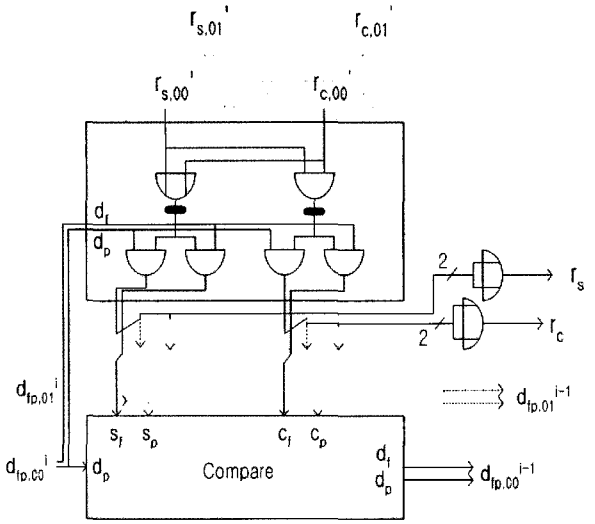

Fig. 7. Another comparator (CSM).

2. Retimed Code Optimized Array

When we decode in forward direction as in Minimized method algorithm, we can use code optimized array presented in [2]. However if we decode

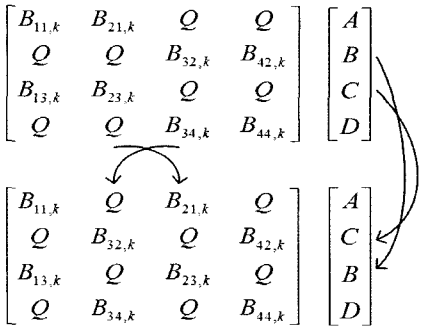

Fig. 8. Switched matrix-vector.

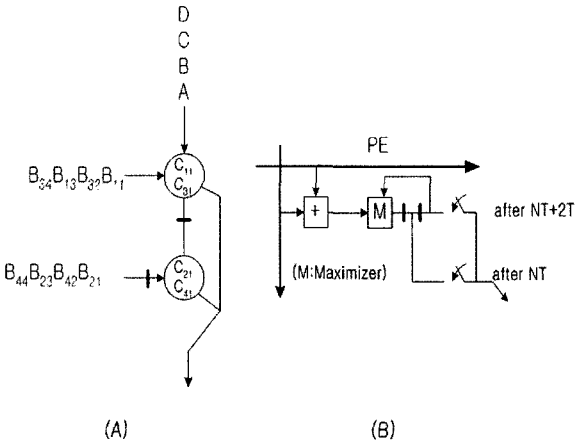

Fig. 9. (A) Proposed code optimized array (B) Its PE.

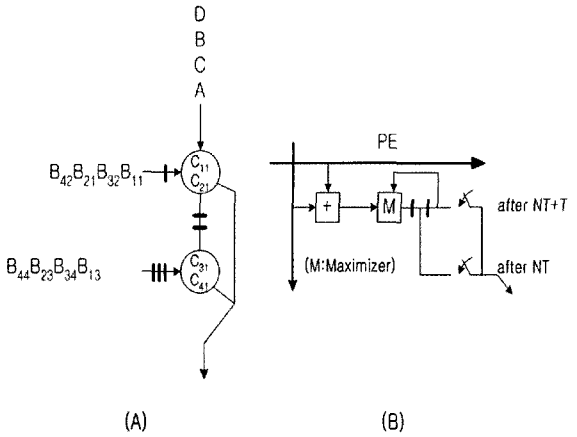

Fig. 10. (A) Modified code optimized array (B) Its PE realization.

in backward realization direction, we cannot apply this scheme. Therefore, we proposed a new code optimized array to apply in backward direction by changing column location of transposed matrix and row location of vector.

IV. Experimental Results

We designed a core block of modified minimized method in bit-level on the basis as in [2, 3, 6~9]. (K=3, R=1/2, 4-bit soft decision) The design is synthesized by Design compiler of SYNOPSYS and TSMC 0.18um library and is timing-simulated by Verilog-XL of CADENCE. The frequency was set in 200 Mhz, 250 Mhz of which slack rate was zero. The results are shown in below graphs. In structure of Fig. 4, we temporarily call the process of finding maximum state (Z_k, Z_{k+2D}) from arbitrary state as 'Level 1'. We show the results of 'Level 1'. When using code optimized array, we need two PEs to calculate new one vector, otherwise we need four PEs. And when K is 3, and survivor depth is 10, to output data in every NT time total of 9 stages are needed for calculating 10 code words. Therefore, we can guess that the area will be reduced about 46,000 gates (9*2*2602) in 9 stages by applying code optimized array to each PE in backward direction. Actually, we implemented the block of 'Level 1' in backward direction to prove that without using code optimized array. As a result, we got 35,000 gates difference. Also, when we used comparator of [5], area was smaller than using that of [2].

V. CONCLUSIONS

By using the code optimized array in backward direction of minimized method algorithm, we could reduce the area complexity which is smaller than that of forward method. Also, we could reduce the area complexity by utilizing a radix-2 comparator which was converted from radix-4 comparator. Furthermore, while synthesis frequency was 250 Mhz, the new comparator exhibits a more suitable timing simulation. As a future work, we believe that there is room to reduce more area and wiring for better performance.

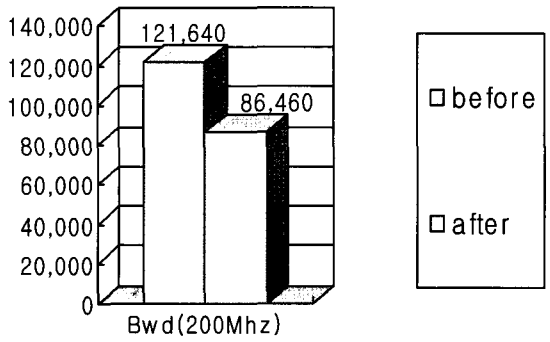

Fig. 11. Area comparison before and after code optimized array(Level 1, Backward).

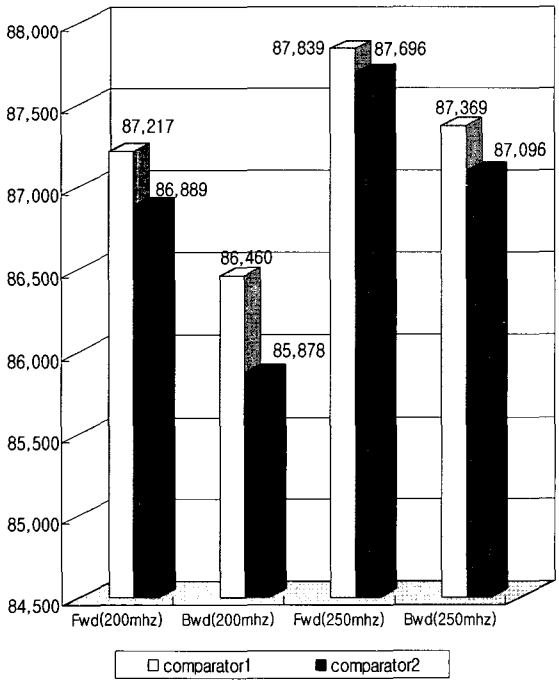

Fig. 12. Area comparison when using another comparator after code optimized array.

ACKNOWLEDGMENTS

This work is partially supported by IT-SoC center, IDEC, KOSEF, and BK21.

REFERENCES

[1] A. J. Viterbi, "Error bounds for convolutional coding and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, vol.IT-13, pp. 260-269, Apr. 1967.

[2] G.Fettweis, H.Meyr, " High rate Viterbi processor : A Systolic array solution," *IEEE J.SAC*, Oct. 1990.

[3] G.Fettweis, H.Dawid, and H.Meyr,"Minimized method Viterbi decoding: 600 Mb/s per chip," *proc. GLOBECOM 90*, vol.3, pp.1712-1716, Dec. 1990.

[4] G.C.Clark, Jr. and J.B. Cain, Error-Correction for Digital Communication. New York : Pleum, 1981.

[5] V. S. Gierenz, O.Weiss, T. G. Noll, I. Carew, J. Ashley, and R. Karabed, "A 550 Mb/s radix-4 bit-level pipelined 16-state 0.25-_m CMOS Viterbi decoder," *Proc. IEEE Int. Conf. Application-Specific Systems,Architectures, and Processors*, pp. 195–201, 2000.

[6] G. Fettweis and H. Meyr, "A 100 Mbit/s Viterbi decoder chip: Novel architecture and its realization," *Proc. IEEE Int. Conf. Commun.*

[7] P. J. Black and T. H.-Y. Meng, "A 1-Gb/s, four-state,sliding block Viterbi decoder," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 6, pp. 797-805, June 1997.

[8] G.Fettweis and H.Meyr, "Feedforward architecture for parallel Viterbi decoding," *J.VLSI Signal Processing*, vol.3, pp.105-119, 1991.

[9] Je-Hyuk Ryu and Jun-Dong Cho, "Low Power Systolic Array Viterbi Decoder Implementation with a Clock-gating Method," Vol. 12-A No. 1, *Korea Information Processing Society*, pp. 1-6, Feb 2005.

Min Woo Kim received the B.S. degree in electronic and electrical engineering from Sung Kyun Kwan University, Suwon, Korea, in 2005 where he is currently working toward the M.S.degree in electronic and electrical engineering. His research interest includes high speed viterbi decoder.

Jun Dong Cho received M.S degree from Polytechnic University, Brooklyn, NY, 1989, and Ph.D. degree from Northwestern University, Evanston, IL, 1993, both in computer science. He was a Senior CAD Engineer at Samsung Electronics, Co., Ltd. He is now Professor of Electronic Engineering Dept., Sungkyunkwan University, Korea. His research interests are in the area of VLSI/SoC CAD algorithms and lower power communication and multimedia system designs. He received the Best paper award at the 1993 Design Automation Conference. He has been a guest editor of VLSI DESIGN for the several special issues in CAD and VLSI designs. He is a IEEE Senior Member.