High speed area efficient configurable Viterbi decoder for WiFi and WiMAX systems

Sridhar Nandula, Yepuri Sudhakara Rao, and Siva Prasad Embanath

Abstract— **In this paper, an area efficient configurable design for high speed Viterbi Decoder suitable for IEEE 802.11 based wireless LAN and IEEE 802.16e based WiMAX has been proposed. This design also supports the puncturing schemes defined in the above wireless standards. An area efficient VLSI design for trace back unit has been proposed in this paper. Synthesis results targeting FPGA and ASIC are included. These results show that the new architecture can achieve good speed, while offering significant area advantage.**

***Index Terms*—ACS, ASIC, BER, BMG, FPGA, Wi-Fi, WiMAX**

I. INTRODUCTION

Latest wireless communication technologies like Wi-Fi and WiMAX are opening up new challenges in baseband hardware design and demanding high speed, area efficient and reconfigurable designs. As the data speeds are moving skywards, demand for high speed Viterbi decoders is increasing. Convolutional coding with Viterbi decoding is a popular and powerful method for Forward Error Correction (FEC) in communication systems. The proposed design particularly optimized for high speed with out compromising on performance so that it suits the requirements of latest wireless standards like 802.11abg and 802.16de.

This paper has been divided into 8 sections. Section 4 discusses about an efficient ACS unit design [3]. Section 5 discusses about metric normalization method used in this design [3]. Section 6 discusses about the trace back unit design and its optimization that enables area efficient design. Section 7 discusses ASIC and FPGA synthesis results.

II. VITERBI DECODING ALGORITHM

The Viterbi decoding algorithm is widely used in both wired and wireless communication to derive maximum likelihood sequence estimates of transmitted data in noisy channel environment. This algorithm uses trellis description of the channel outputs. It recursively computes for each state of the trellis the best fit for the received sequence among the trellis sequences which end in the specific state. These best fit sequences are stored in a memory and traced back to find the best possible decoding sequence. These best fit sequences are called survivor sequences.

The Viterbi algorithm is outlined as follows.

1. At each time unit (*j*), compute branch metrics for each possible path between the states
2. Add these branch metrics to the cumulative path metric value (*j-1*)
3. Compare two path metrics entering each state and select the path that is entering the state with minimum path metric and store the survivor path
4. Repeat this procedure for all the states and store the survivor paths at each state
5. Repeat above procedure until the trace back depth is reached
6. Once the trace back depth is reached, find the minimum of the survivor path metrics and use that path to trace back and to decode the information bits along that path

Recent wireless broadband standards like 802.11abg [5] and 802.16de [6], [7] uses a rate ½ Convolutional encoder with constraint length K=7. The generator polynomials used are $G_1=171_{(o)}$ and $G_2=133_{(o)}$. Both these standards also support puncturing rates 1/2, 2/3 and 3/4. We implemented a high speed area efficient configurable Viterbi decoder to support both these standard requirements.

III. BRANCH METRIC GENERATOR

The Branch Metric Generator (BMG) computes the branch metrics for each symbol of the input sequence by calculating the hamming distance between the input symbol and the expected symbol for each connection of the trellis. For example, for a rate ½ code, the possible symbol combinations in the encoded sequence are: 00, 01, 10 and 11.

However, instead of Hamming distance, Euclidean distance may be used to calculate the branch metric. If the demodulated symbols are directly decoded as single bits they are treated as “hard” symbols. In other words, if the demodulated symbols

Manuscript received June 18, 2007.

Sridhar Nandula is working as Engineering Manager in a semiconductor company. Phone: 91-9953330686; e-mail: nandulasridhar@gmail.com
Yepuri Sudhakara Rao is a Masters degree student at IIT Madras, India. e-mail: yepuri@gmail.com
Siva Prasad Embanath is working as Senior Engineer in Singapore. e-mail: sivaprasade@gmail.com

are quantized to 3 or 4 bit values, they are treated as "soft" symbols. Fig 1 shows the branch metric diagram for 8 levels i.e., with 3 bit quantization. In our implementation we used 4 bit quantization for quantizing the encoded symbols. Hence, the transmitted bits in a symbol were represented as 16 possible levels of signals. The quantization level affects the way the branch metrics are generated. By increasing the number of soft bits (quantization factor) the number of possible values for branch metrics also increases.

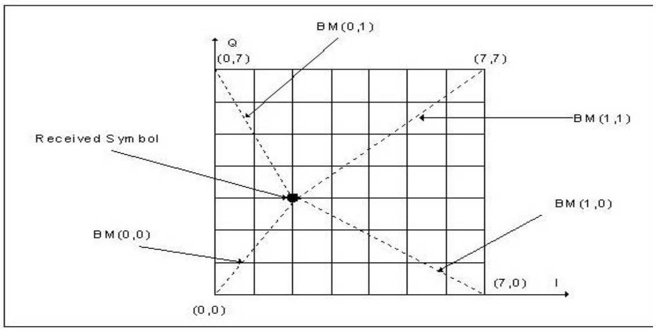

Fig. 1. Soft decision based Branch metric generation

Different measures such as Euclidean distance or Manhattan distance can be used to calculate the branch metrics. Eq 1 shows the equation for Euclidean distance.

$$BM_e = (X_r - X_e)^2 + (Y_r - Y_e)^2 \qquad (1)$$

where Xr and Xe are received and expected symbols respectively. In this paper, we have used Manhattan distance to generate branch metrics. Eq 2 shows the equation for Manhattan distance.

$$BM_e = |X_r - X_e| + |Y_r - Y_e| \qquad (2)$$

In hardware, computation of Manhattan distance is easy when compared to Euclidean distance. In our studies we observed that the performance loss with Manhattan distance over Euclidean distance is around 0.2dB.

IV. ACS UNIT

Add, Compare and Select is one of the important component that is responsible for speed of the operation. Each node or state corresponds to one ACS unit. The constraint length *K* decides number of ACS units needed for the Viterbi algorithm. The maximum number of ACS units needed is given by 2^{K-1}. Based on the targeted operating frequency, these ACS units can be shared to reduce the total hardware occupied on the device. However, to maximize speed of operation, one needs to use 2^{K-1} ACS units (one for each state). In this implementation, we have used 2^{K-1} ACS units (64) to achieve maximum speed.

Some efficient VLSI architectures were proposed [3] for ACS unit design. This design uses the Modulo Normalized ACS with Modified Comparison Rule. Fig 2 shows digital hardware diagram for this ACS unit.

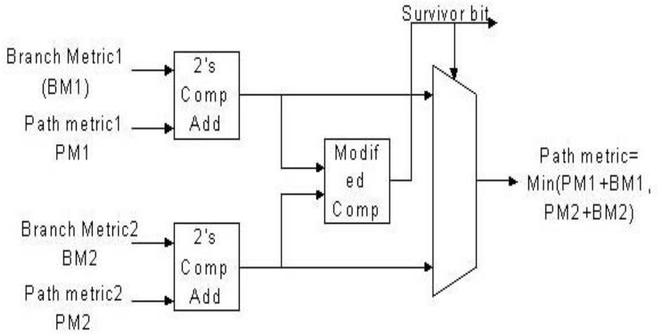

Fig. 2. Architecture for Modulo Normalized ACS with Modified comparison rule

V. METRIC NORMALIZATION

If the encoded data is a continuous stream of bits, the path metric which is the accumulated value of the branch metrics, may overflow. In order to avoid such scenario, we need to normalize the path metrics at regular intervals or apply a normalization technique that automatically normalizes the path metric. The modified comparison rule [3] discussed above can do this metric normalization automatically.

In the present design, both Modulo Normalization Technique and Modified Comparison Rule are used to rescale the path metrics. In modular arithmetic the metric m_j is replaced by the normalized metric $m_{norm} = (m_j + c/2)\ (modC) - C/2$ where C is the maximum value represented by the number of bits used for defining path metric. It was found that the differences in the survivor metrics remain unchanged using modular arithmetic provided that $\delta <= C/2$ where δ is the bound of the difference between survivor metrics. For hardware consideration 2's complement adder can do this normalization, after normalization we can use modified comparison rule which gives the survivor bits without an error in the case of overflow. The Modified Comparison Rule is given by:

$$Survivor(m_{norm1}, m_{norm2}) = m_{norm1} xor \ \ m_{norm2} xor \ \ y(m_1, m_2) \qquad (3)$$

where $y(m1, m2)$ denotes the unsigned comparison. In other words survivor (m_{n1}, m_{n2}) equals $y(m1, m2)$ or logical inverse of $y\ (m1, m2)$ if m_{n1} and m_{n2} have the same (or opposite) sign.

VI. TRACEBACK UNIT

This is the most important design unit in Viterbi decoder. The important characteristic is that every state from a current time is followed backwards through its maximum likelihood path, all the paths converge at a point somewhere previous in time the point where the corrected bit streams starts. This point is called the merger point or trace back depth. This is how trace back decisively determines the state of the encoder at a given time. It is proved that the Viterbi algorithm always follows the maximum likelihood path.

The performance of Viterbi decoder largely depends upon the trace back depth. The larger the depth of the trace back, the better the performance is. However, the increase in trace back depth increases the hardware complexity exponentially so one has to trade off between the performance level and the complexity of the hardware. From literature we can fix trace back depth = 5*constraint length for non - punctured codes. However, for punctured codes like rate 2/3, 3/4, 8*constraint length is recommended. In this implementation we studied the performance of Viterbi decoder for different trace back depths.

A systolic architecture has been proposed [4] that uses *register exchange method* to exchange the survivor path information between the states. This systolic architecture register exchange method is very efficient for high speed Viterbi decoders and need not trace back full memory for each decoded bit. Fig 3 shows the basic systolic architecture for trace back unit.

However, this architecture demands extensive set of registers to store the survivor path. In this paper, we propose a modified systolic architecture for trace back unit that reduces the number of registers needed for the same trace back depth. This will reduce significant hardware without compromising on performance. When the available area on the device is limited, this trace back greatly helps in reducing the over all size of the design.

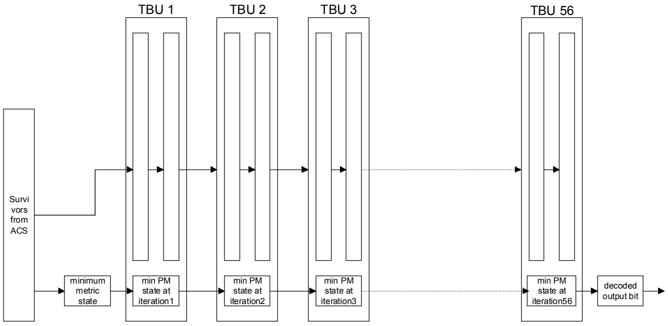

Fig 3. Trace back unit architecture

In the basic implementation, we considered 56 (8*k) as trace back depth. This trace back unit consists of 112 registers each of 64-bit and 56 registers each of 6-bits to store the lowest pick states. At every stage, we need to push the survivor bits found from ACS unit from one end of the trace back and keep shifting the other survivors to the next stage. Simultaneously, it takes the lowest pick (from the lowest pick unit) and also updates the state registers. Once the trace back is completely filled, the MSB of the state register will be the decoded output.

In the modified trace back implementation, some of the intermediate registers are skipped and the survivor path information for the next node is pre calculated by using combinational logic. This reduces the number of shift registers used for the trace back at the expense of combinational logic.

VII. IMPLEMENTATION RESULTS

In this work, the proposed modified trace back unit Viterbi decoder and the basic high speed Viterbi decoder were coded in Verilog RTL. This Verilog RTL implementation is verified with the test vectors generated from the C model prototype. These RTL implementations were synthesized for 130 nm ASIC library and Xilinx Virtex 2 FPGA device. Table 1 shows the synthesis results.

	Area		Speed (in MHz)	
	FPGA (in slices)	ASIC (in Kilo gates)	FPGA	ASIC
Basic trace back Viterbi	6424	131.4	100	200
Area efficient trace back Viterbi	5261	125.7	71	160

Synthesis results clearly shows that by removing some registers in trace back and introducing the combinational logic decrease the device utilization. However, the cost of this reduction is reduction in frequency of operation. This is not a limitation for Wi-Fi and WiMAX cases as the implemented design can meet the through put requirement (around 70 mbps) by a big margin.

VIII. CONCLUSIONS

A high speed and area efficient Viterbi decoder VLSI architecture has been proposed. The modification in trace back unit design reduces the area utilization for the same error decoding performance. Proposed design can be synthesized for ASIC or FPGA with out any modifications. This design can be used for WiFi and WiMAX baseband receivers. In case of higher puncturing rates, the BER may be worse. By increasing the depth of traceback we can compensate for this loss.

REFERENCES

[1] Bernard Sklar "*Digital Communications: Fundamentals and Applications*", 2nd Ed. Printice Hall.

[2] John B. Anderson and Seshadi Mohan, "*Source and channel coding – An Algorithmic Approach*", chapter 4 1991, Kluwer Academic Publisher.

[3] C. B. Shung, P. H. Siegel, Gottfried Ungerboeck and Hemant K ThaparT. K. Trung, Ming Tang Shih, Irving S Reed and E. H. Satorius "*VLSI architectures for Metric Normalization in the Viterbi algorithm design*", IEEE Transaction on Communications, vol.40, No.3, March 1992, p.616-624.

[4] T. K. Trung, Ming Tang Shih, Irving S Reed and E. H. Satorius "*A VLSI design for a Traceback Viterbi Decoder* ", IEEE Transaction on Communications, vol.40, No.3, March 1992, p.616-624.

[5] IEEE Std. 802.11a-1999, *"Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: High-speed physical layer in 5GHz Band,"* 1999 edition.

[6] IEEE Std. 802.16d-2004, *"Air Interface for Fixed Broad Band Wireless Access Systems,"* 2004 edition.

[7] IEEE Std. 802.16e-2005, *"Air Interface for Fixed and Mobile Broad Band Wireless Access Systems,"* 2005 edition.

[8] M. Boo, F. Arguello, J. D. Bruguera, R. Doallo, and Zapatta "*High performance VLSI Architecture for the Viterbi Algorithm*", IEEE Transactions on Communications, Vol. 45, No.2, February –1997