A SYSTOLIC ALGORITHM FOR VLSI DESIGN OF A 1/N RATE VITERBI DECODER

Sadiq M. Sait Ali F. Damati Mushfiqur Rahman
King Fahd University of Petroleum and Minerals
KFUPM #673, Dhahran-31261, Saudi Arabia

Abstract: Viterbi decoding algorithm is one of the most widely used forward error-correcting techniques in digital communication. Hardware realizations of the Viterbi algorithm are complex, and implementation is difficult, expensive and/or slow. Systolic architectures are simple, modular and regular and are well suited for VLSI implementation. In this paper a new systolic architecture for Viterbi decoding is presented. It consists of two blocks of processors. The first contains a column of processors which perform branch metric computation and decide on the survived branches. The second consists of a matrix of simpler processors which update survived paths and provide the decoded output. The systolic algorithm is modelled in AHPL to verify functional correctness. Implementation details are discussed. The proposed architecture is compared with previous implementations of the Viterbi algorithm.

1. INTRODUCTION

Error-correction coding used in digital data communication improves the reliability of communication channels. In convolutional codes the output is determined not only by the current input block, but also by the (m-1) previous input digits where m is the *constraint length* of the code.

Performance of convolutional codes has been proved to be equal or superior to that of block codes [BHAR81, CLAR81]. Also the encoder implementation for convolutional codes is simpler. Viterbi decoding algorithm [VITE71] yields maximum likelihood decisions [FORN73], and is particularly desirable for efficient communications at very high data rate where large decoding delays are intolerable. It is a practical method for improving the efficiency of satellite and space communications [HELL71], due to the fact that channels available for space communication are frequently modeled as white Gaussian channels with adequate accuracy [HELL71]. The Viterbi algorithm has been proved to provide performance superior to all other coding schemes with considerable robustness against varying channel parameters [HELL71]. The draw back of Viterbi decoding, however, is the complexity of the decoder hardware which is due to the large storage required. Also computation in the decoding process increases exponentially with the constraint length of the code.

One of the first Viterbi decoders used in a practical communication system was the Linkabit LV7026 which required 356 integrated circuits. More recently several microprocessor implementations for the Viterbi decoder have been proposed [CONA76, RASH82, SAID86]. With the recent developments in VLSI technology, several authors have proposed VLSI architectures for the Viterbi decoder [FREN86, GULA86]. The various tradeoffs in the design and implementation of practical Viterbi decoders are complexity of the design, maximum operating speed and size of the circuit.

Kung introduced the concept of *systolic arrays* [KUNG79]. Systolic arrays are a methodology for combining logic and memory to create powerful *simple regular structures*. Systolic arrays consist of a network of simple processors (cells) that circulate data in a regular fashion so that processor and communication resources can be fully utilized. Data and control flow in the array is usually simple and regular, so that communication among cells is local. Long distance or irregular communications are usually avoided. Every processor in the array *rhythmically pumps data in and out*, each time performing some short computation, so that a regular flow of data is kept up in the system. One important feature of the systolic arrays is that the array uses extensive pipelining and multiprocessing so that a large proportion of the processors in the array are kept active which results in sustaining a high rate of computation flow. More details about the realization of algorithms in hardware using systolic arrays can be found in [KUNG79].

Viterbi algorithm can be represented by a sequence of elementary operations which are performed in a repetitive manner. A similar observation was made by Clark and Cain [CLAR81]. We would like to exploit this observation and design a systolic decoder for the Viterbi algorithm.

In this paper we present a systolic architecture for the Viterbi decoding algorithm for rate 1/n convolutional codes. Section 2 first reviews the Viterbi algorithm briefly and then presents the systolic design. Implementation results of the processors are presented in Section 3. Extensions to rate k/n decoders are presented in Section 4 followed by concluding remarks in Section 5.

2. SYSTOLIC ARCHITECTURE OF VITERBI DECODING

In this architecture the decoding process is divided into two blocks as shown in Fig.1. For a **k/n** rate convolutional code, the first block (**block**$_1$ or **path metric block**) consists of a column of r^{m-k} identical processors which compute path metrics. Each path metric block processor (**PROC**) consists of **r** registers, **r** adders, and **r** flags, where **r** indicates the number of possible transitions from any state in the trellis structure [CLAR81]. In this section the focus will be on the binary trellis (**r**=2) and 1/2 rate convolutional code with constraint length **m**=3.

The internal architecture of a processor of the first block is shown in Fig.2. It consists of two registers R_1 and R_2, two Hamming distance calculators, two adders which add the previous path metric to the Hamming distance between the input block and the branch constant and a comparator which compares the outputs of the adders and sets the corresponding flag F_1 or F_2.

The second block (**block**$_2$ or **path history block**) stores the path in the trellis; it consists of a mapping of the trellis into a matrix of simple processors where each processor (**P**) corresponds to a node of the trellis. The processors in the second block are connected in such a way that in each clock period, depending on the survived branches, the paths corresponding to each state are updated.

Each processor $P_{c,n}$ stores 2 bits, and has 4 inputs and 4 outputs as shown in Fig.3. The inputs designated as F_{1_i} and F_{2_i} are connected to the outputs F_{1_o} and F_{2_o} of the processors $P_{c,n-1}$ to the right in the same row. The inputs designated as q_{1_i} and q_{2_i} are connected to the outputs q_{1_o} and q_{2_o} of two other processors in the right adjacent column whose address is given by the left shuffle and left shuffle exchange of the address of processor $P_{c,n}$, as shown in Fig.1. The function of the processor $P_{c,n}$, $(0 < c \leq i,\ 1 \leq n \leq 2^{m-1})$ is as follows:

```
case
  (q1i + q2i = 0 ): F1o ← F2o ← q1o ← q2o ← 0;
  otherwise:
      begin
         F1o ← q1o ← F1i
         F2o ← q2o ← F2i
      end
endcase
```

Transfer of contents of processors in column **i** of the trellis structure, to those in column **i+1** (on the left) takes place at every clock pulse, and the history of survived paths is stored. Contents of flags in processors of the second block indicate the survived paths of the trellis.

The bits received by the decoder will in general be different from the transmitted bits represented by the branch labels of a codeword path in the trellis [CLAR81]. A maximum likelihood decoder is required to find a path in the trellis which matches most closely to the received sequence. As the path length increases, the number of comparisons grow exponentially. Viterbi decoding algorithm [FORN73] essentially performs maximum likelihood decoding; however, it reduces the computational load by taking advantage of the special structure of the trellis. The algorithm involves calculating the Hamming distance between the received signal, and all the trellis paths entering each state at time t_i. When two paths enter the same state, the one having the best metric is chosen; this path is called the *surviving path*. The selection of surviving paths is performed for all states. The decoder continues in this way to advance deeper into the trellis, making decisions by eliminating the least likely paths. Furthermore, all the surviving paths start merging after some time and the decoder output is simply the bit corresponding to this branch common to all the survived paths.

The output of the decoder is a function of the leftmost column of the second block of processors. If all the 2^{k-1} paths have merged at the left most column of the second block, then the flags F_1 and F_2 of all processors except one will be zeros. The corresponding flags of the node from which all the survived paths emanate will have F_1 and F_2 equal to 1 and 0, or, 0 and 1, depending on whether the decoded output is 0 or 1. The proposed design is based on the assumption that the survivor paths have merged in at least the left most branch, and only one processor in the last column will have a non-zero flag F_1 or F_2. The decoder output can be obtained by ORing the F_2 flags of all processors of the last column.

3. IMPLEMENTATION, SIMULATION AND VALIDATION

The functional correctness of the algorithm has been verified by modelling the decoder [DAMA88] in AHPL and simulating the model with the help of the AHPL functional level simulator [ALSH83]. SPICE simulation for MOS circuits of basic cells of both types of processors have been carried out in order to make sure that the design is error-free and also to estimate the speed of the systolic decoder. HDCs are implemented using PLAs and the comparator is designed using a multiplexer. The transient analysis showed that a 1/2 rate systolic decoder can operate up to 12 Mbits/s.

The area of processors has been determined by mapping the circuits to MOS VLSI layouts [DAMA88]. The data format and language chosen for MOS VLSI design is LUCIE. Design rules of IMAG/TIM3 are used in the design of layouts [GUYO]. Layouts of various cells such as full-adder, multiplexer, inverter, pass transistors are

first made. These cells are then interconnected to form the required processors. The design rules of individual cells and the entire layout were checked by IMAG/TIM3 using Silvo Liascos DR checker. The area of **PROC** is $1350*750\lambda^2$ and that of **P** is $280*230\lambda^2$

4. EXTENSION TO RATE K/N CODES

In this section we show how the systolic design for a rate 1/2 code can easily be adapted for a general rate **k/n** code. The modifications required for rate **R=1/n** code are quite simple since the same encoder structure is retained; only the number of mod-2 adders is increased from 2 to **n**. Increasing the number of mod-2 adders causes an increase in channel symbols for each branch of the trellis structure from 2 to **n**. The architecture of 1/**n** rate systolic decoder is identical to that of 1/2 rate decoder except that an appropriate increase in number of bits of HDCs, adders, comparators and registers are needed.

The VLSI realization becomes complex for larger values of **k** because the number of HDCs, adders etc. grows exponentially, i.e., $O(2^k)$, thereby presenting serious implementation difficulties.

The key to simplifying the implementation is to use the so called "punctured convolutional codes" [CAIN79]. The new code takes a **R=1/n** code and "punctures" or deletes certain channel symbols thereby producing a **R=k/n** code. The practical value of the punctured code approach is obvious. Therefore, one can implement an **R=2/3** decoder as an **R=1/2** decoder with additional control to stuff erasures in the locations of the deleted bits. After the erasures are stuffed, decoding proceeds just as if the code were an **R=1/2** code.

5. CONCLUDING REMARKS

A new systolic architecture for a Viterbi decoder is presented. The design consists of simple processors and provides regular and modular layouts, suitable for VLSI implementation. The decoding depth, that is, the length of the survived paths, can easily be extended by cascading two or more path history blocks ($block_2$) fabricated on separate chips. This is a powerful feature which is desirable, since the optimum decoding depth depends on the code, channel conditions, allowable decoding delay etc. The proposed design allows flexible decoding depth (in multiples of the number of $block_2$) while for the other known hardware designs [FREN86, GULA86] the decoding depth can only be changed by modifying the basic design.

The proposed systolic design compares favorably against other known implementations of Viterbi decoders, in terms of speed and modularity. In [SAID86] a software implementation of the Viterbi algorithm on the MC68000 microprocessor was proposed. In [CONA76] a general class of machines based on micro/mini computers was presented. Although the software implementations have some inherent flexibility, they are unable to support high data rates. Also, because of lack of parallelism, the maximum data rate which can be supported falls drastically as the constraint length is increased.

A recent implementation of Viterbi decoder for a digital communication system with a time dispersive channel has been proposed in [FREN86]. It is designed for a rate 1/2 convolutional code with constraint length **m**=10, consists of 5 chips. If a comparison is made between this design and the systolic decoder, we find that there is an area-speed tradeoff between the two designs. The systolic decoder is designed to operate at high data rates, up to a maximum of 12 Mbits/s while the other design is meant for data communications over voice channels at 2400 bit/s. For the same specifications the area occupied by design in [FREN86] is lesser, however for small constraint length, the systolic decoder can be implemented on a single chip.

The systolic design of the Viterbi decoder presented here consists of simple basic processors with high array efficiency and high allowable data rates. Comparison between the proposed design and other known designs indicates significant advantages in the areas of speed, size and power dissipation, which makes it a suitable candidate for high speed real-time applications.

Acknowledgment: The authors acknowledge the support of King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia.

REFERENCES

[ALSH83] M.M.Al-Sharif, "Functional Level Simulator for Universal AHPL", M.S.Thesis, University of Arizona, 1983.

[BHAR81] V.K. Bhargava, D. Haccoun, R. Matyas, P. Nuspl, Digital Communications by Satellites, John wiley & Sons, New York 1981.

[CAIN79] J.B Cain, G.C. Clark, JR., J.M. Geist, "Punctured convolutional codes of rate (n-1)/n and simplified maximum likelihood decoding," IEEE Trans. Inf. Theory, vol. IT-25, No. 1, Jan. 1979.

[CLAR81] G.C. Clark and J. Cain, Error-Correction Coding for Digital Communications, Plenum Press 1981.

[CONA76] J. Conan and R. Oliver, "Hardware and software implementation of the Viterbi decoding algorithm for convolutional codes," Proc. ISMM '76., Nov. 1976, Canada.

[DAMA88] F.A. Damati, "A systolic algorithm for VLSI design of a Viterbi decoder", M.S.Thesis, King Fahd University of Petroleum and Minerals, 1988.

[FORN73] G.D. Forney, Jr., "The Viterbi algorithm," IEEE Proc., vol. 61, No.3, pp. 268-278, March 1973.

[FREN86] N. Frenette, P. McLane, L.

Peppard, F. Cotter, "Implementation of a Viterbi processor for a digital communications system with a time- dispersive channel," IEEE Journal on Selected Areas in Common., vol. SAC-4, No.1, pp. 160-167, Jan. 1986.

[GULA86] P.G. Gulak and E. Shwedyk, "VLSI structures for Viterbi Receivers: part I - General theory and applications," IEEE Journal on Selected Areas in Commun, vol. SAC - 4, NO.1, pp. 142-154, Jan 1986.

[GUYO] A. Guyot, A. Jerrage and J. Raymond, "Language Universitaire Conception de Circuits Integres pur 1' Enseighement," IMAG report.

[HELL71] J.A. Heller and I.M. Jacobs, "Viterbi decoding for satellite and space communication," IEEE Trans. Commun. Technol., vol. COM-19, pp. 835-847, Oct. 1971.

[KUNG79] H.T. Kung, "Let's design algorithms for VLSI systems," Proc. Caltech conf. on VLSI, pp. 65-90, Jan. 1979.

[RASH82] H.F. Rashvand, "Implementation of microprocessors in trellis decoding of convolutional codes," Electronic letters, vol18, No. 3, pp. 121-123, Feb. 1982.

[SAID86] S.M. Said and K.R. Dimond, "Real-time implementation of the Viterbi decoding algorithm on a high-performance microprocessor," Microprocessors and Microsystems, vol. 10, No.1, pp. 11-16, Jan/Feb.1986

[VITE67] A.J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," IEEE Trans. Inf. Theory, vol. IT-13, pp. 260-269, April 1967.

[VITE71] A.J. Viterbi, "Convolutional codes and their performance in communication systems," IEEE Trans. Commun. Technol., vol. COM-19, pp. 751-772, Oct. 1971.

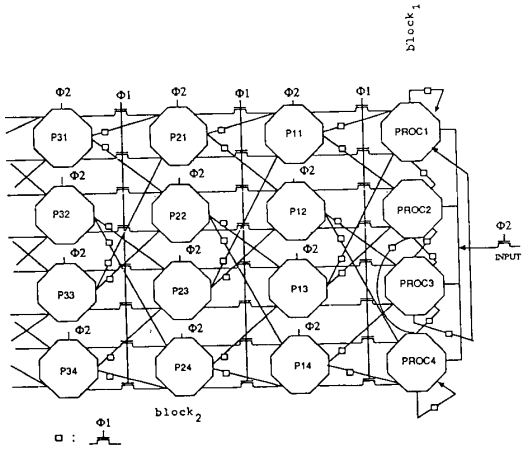

Fig.1. Interconnection between processors of **block**$_1$ and **block**$_2$

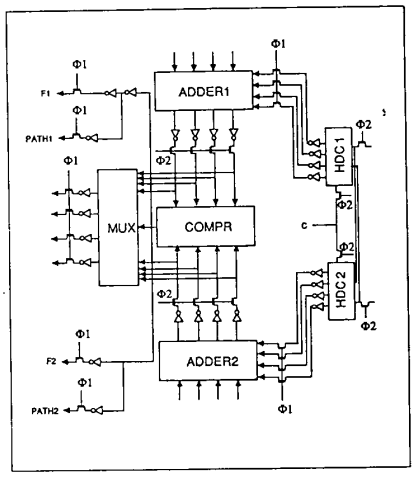

Fig.2. Detailed MOS circuit for processor of **block**$_1$ **(PROC)**

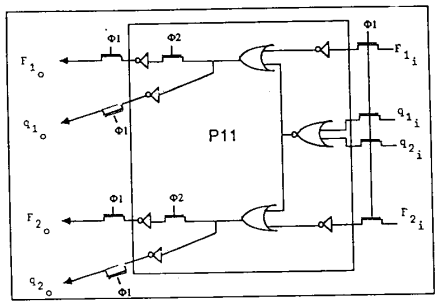

Fig.3. Internal architecture of processor of **block**$_2$ **(P)**