© IEE 1999 6 October 1999
Electronics Letters Online No: 19991411
DOI: 10.1049/el:19991411

Yu Zhang, Ming Dai, Yiman Hua and Gonghuan Du (*Institute of Acoustics, State Key Laboratory of Modern Acoustics, Nanjing University, Nanjing 210093, People's Republic of China*)

E-mail: zhudu@nju.edu.cn

References

1 SHORT, K.M., and PARKER, A.T.: 'Unmasking a hyperchaotic communication schemes', *Phys. Rev. E*, 1998, **58**, pp. 1159–1162
2 PEREZ, G., and CERDEIRA, H.: 'Extracting messages masked by chaos', *Phys. Rev. Lett.*, 1995, **74**, pp. 1970–1973
3 KOCAREV, L., and PARLITZ, U.: 'General approach for chaotic synchronization with application to communication', *Phys. Rev. Lett.*, 1995, **74**, pp. 5028–5031
4 ZHANG, Y., DAI, M., HUA, Y.M., NI, W.S., and DU, G.H.: 'Digital communication by active-passive-decomposition synchronization in hyperchaotic systems', *Phys. Rev. E*, 1998, **58**, pp. 3022–3027
5 DAI, M., ZHANG, Y., HUA, Y.M., NI, W.S., and DU, G.H.: 'Implementation of secure digital communication by hyperchaotic synchronisation', *Electron. Lett.*, 1998, **34**, pp. 951–953
6 ROHATGI, V.K.: 'An introduction to probability theory and mathematical statistics' (John Wiley & Sons, New York 1976), Chap. 12

Implementation of high speed Viterbi detectors

T. Conway

The normal Viterbi architecture and a radix 4 architecture are compared with a parallel ACS version using a latch based storage element. Results obtained using an eight-state EPR4 Viterbi detector show that parallel ACS architectures can provide a high level of performance with modest area requirements.

Introduction: The Viterbi algorithm has been used widely in communications systems as a maximum likelihood sequence detector for decoding convolutional and trellis codes as well as detecting data in a channel with ISI [1]. In this Letter the hardware implementations for high speed communications/storage applications are considered. In particular, the example of an eight-state detector based on the EPR4 partial response used in disk drive storage applications will be used. The algorithm is normally implemented in three blocks, a branch metric unit, an add compare select (ACS) block, and a survivor path memory. For high speed operation, the ACS block becomes the limiting factor as it requires a recursive calculation that is difficult to pipeline. The basic ACS operation requires branch metrics λ_{AC}^{k} and λ_{BC}^{k} to be added to path metrics Γ_A^k and Γ_B^k, where *k* represents a time index. A comparator determines which sum is the largest and a multiplexer then selects that value as the value of Γ_C^{k+1}. The number of ACS units required is equal to the number of states in the detector. The critical path in such an ACS unit will be the concatenated delay of an adder, a comparator, and a selector leading to the term 'add, compare, select' bottleneck. Each state requires two adders, one comparator and a two-way multiplexer.

Radix 4 ACS: For higher speed operation a radix 4 implementation can be designed which operates on two samples per clock cycle [2]. As the normal ACS unit selects the maximum between two values, the radix 4 ACS must add four branch metrics and select the maximum between the four values. To maximise the speed of operation, the maximum of four values can be determined by using six parallel comparisons. The speed of operation would then be determined by the delays due to the addition, the comparison and the select operations. Noting that two input samples are processed per cycle, the throughput can potentially be almost doubled. However, each state now requires four adders, six comparators and a four-way multiplexer. Also, as two samples are processed per cycle, the branch metric unit needs to generate many more branch metrics than required in the normal ACS implementation. Hence, the improved speed of operation is achieved at the cost of a significant hardware overhead.

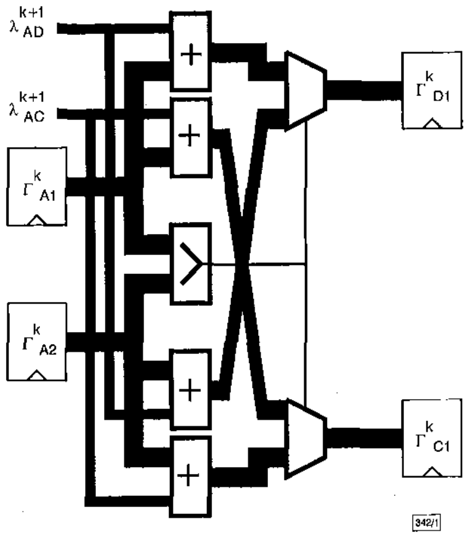

Fig. 1 *Parallel ACS unit*

Parallel architecture: The parallel architecture replaces each path metric register with two registers which store the sum of the branch metric and the appropriate branch metric as suggested in [3]. Hence, registers are used to store $\Gamma_A^k + \lambda_{AC}^k$ and $\Gamma_B^k + \lambda_{BC}^k$. During each clock cycle, these are compared to make a decision. Simultaneously, these values have the two possible metrics for the next cycle added and the correct metric is chosen when the comparison is completed. In this way, the compare and addition operations are carried out in parallel rather than serially resulting in an increase in throughput. Fig. 1 shows the parallel ACS structure.

Register Γ_{A1} holds the value $\Gamma_A^k + \lambda_{AC}$ while Γ_{A2} holds the value $\Gamma_B^k + \lambda_{BC}^k$. The improvement in speed is achieved at the expense of an additional path metric register, two branch metric adders and a multiplexer for each state of the detector. While these additional hardware elements increase the area of the detector, they should not increase the area as much as a radix 4 architecture.

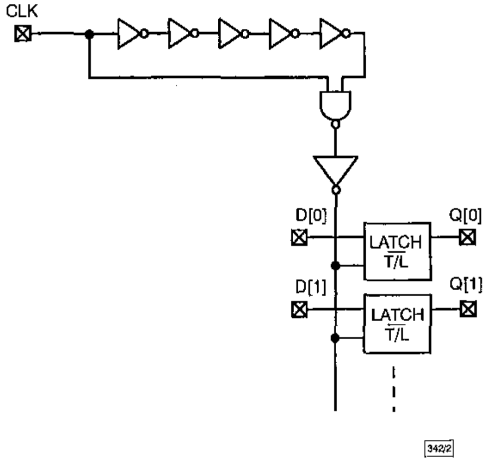

Fig. 2 *Latch-based storage element*

Storage element: As the critical path delay is now of the order of a single adder and multiplexer, the actual register setup and propagation time become a significant fraction of the total cycle time. Using the EPR4 example, the cell library edge triggered flip-flop setup and propagation times accounted for 30% of the cycle time. Although a full custom flip-flop could perform better, a level triggered latch from the cell library was used which had setup and propagation times 70% that of the flip-flop cell. Use of the circuit in Fig. 2 as the register element for the path and branch metrics enabled a 10% overall increase in the operation speed of the detector to be obtained. The operation of Fig. 2 implies that the latch is transparent for a short period of time, and hence the logic in the ACS unit must have a lower limit on its propagation time. This is easily achieved using the parallel ACS unit of Fig. 1.

EPR4 detector: As an example of the proposed structure, a Viterbi detector for the EPR4 partial response of $1 + D - D^2 - D^3$ was designed using the two existing structures and the proposed structure. The EPR4 Viterbi detector requires eight states and high speed implementation is of interest due to its use in PRML read channels [4]. For the detector implementation, the input signal is quantised to 6 bit resolution using two's complement binary numbers. The branch metrics are represented using 7 bit unsigned numbers and the path metrics are represented with 9 bits two's complement numbers. Metric overflow is handled using the modulo nature of the two's complement numbers used for the path metrics [5].

For the EPR4 partial response there are five possible signal levels and hence five branch metrics need to be calculated. However, the radix 4 based detector requires 17 branch metrics as it operates on two-input samples per cycle.

Results: The three methods for implementing the ACS units were used to implement a full eight-state EPR4 detector including branch metric calculations and survivor path memory with depth 32. The logic for each was synthesised from a hardware description language version of each method, targeting a 0.7μm CMOS two-layer metal cell library. The synthesised designs were placed and routed to extract routing capacitances and calculate the design area. The back annotated designs were simulated to determine the maximum operating rate of each design for the worst case and for typical process parameters.

Table 1: Performance with 0.7μm CMOS cell library

	Area	Clock rate (worst case)	Clock rate (typical)
	μm^2	MHz	MHz
Normal	3576K	65	128
Radix 4	10651K	98	192
Parallel	5724K	91	178

Table 1 shows the simulation results. For the radix 4 architecture, a 50% increase in the worst case clock rate is achievable, but requires almost three times the area of the normal ACS implementation. The speed is not doubled due to the additional fan out and routing required by the complex four-way ACS. The large area increase is due to the ACS complexity and the requirement for significantly more branch metrics to be calculated. The parallel ACS architecture provides an increase of 40% while only requiring a 60% increase in area. This speed increase is close to that for the radix 4 architecture but the additional area required is significantly less.

Conclusions: The parallel architecture of the ACS unit reorders the required calculations by doubling the number of path metric registers while allowing the speed of operation to be increased by allowing the branch metric adders to be completed in parallel with the comparison rather than in series. When this is combined with a latch based metric storage element, a significant improvement in speed compared to the normal ACS architecture is achieved but without the significant area penalty of the radix 4 architecture. The achievable performance was determined through the use of a synthesised EPR4 detector using the parallel and conventional architectures. The results show that it provides a useful point in the area/speed tradeoff available to the designer of high speed Viterbi detectors.

14 September 1999
Electronics Letters Online No: 19991413
DOI: 10.1049/el:19991413

T. Conway (*ECE Department, University of Limerick, National Technological Park, Limerick, Ireland*)

References

1 FORNEY, G.D.: 'Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference', *IEEE Trans.*, 1972, **IT-18**, pp. 363–378

2 BLACK, P.J., and MENG, T.H.: 'A 140-Mb/s, 32-state, radix-4 Viterbi decoder', *IEEE J. Solid-State Circuits*, 1992, **SC-27**, pp. 1877–1885

3 FETTWEIS, G.P., KARABED, R., SIEGEL, P., and THAPAR, H.: 'Method and means for detecting partial response waveforms using a modified dynamic programming heuristic'. United States Patent 5,430,744, July 4, 1995

4 THAPAR, H.K., and PATEL, A.M.: 'A class of partial response systems for increasing storage density in magnetic recording', *IEEE Trans.*, 1987, **MAG-23**, (5), pp. 3666–3668

5 HEKSTRA, A.P.: 'An alternative to metric rescaling in Viterbi decoders', *IEEE Trans.*, 1989, **COM-37**, (11), pp. 1220–1222

Integrated sidelobe energy reduction technique using optimal polyphase codes

W.K. Lee, H.D. Griffiths and R. Benjamin

For the distributed targets arising in most synthetic aperture radar (SAR) images, it is the integrated sidelobe (ISL) energy rather than the peak sidelobe level (PSL) that governs the performance of the pulse compression processing. A sidelobe cancellation technique is suggested which reduces the ISL energy by eliminating a significant portion of the sidelobe pattern of the pulse compression output, while the PSL is preserved at the optimal Barker-code level. The efficiency of this ISL reduction is proportional to the signal code length.

Introduction: In pulse compression processing the quantitative range sidelobe pattern is unknown until the received signal enters the pulse compression processing stage and the output peak value is evaluated. In particular, when the received signal arises from a distributed target scene rather than from a single point target, the precise information is only available in the original raw data, prior to compression, and hence the best way to implement a sidelobe canceller is to extract the relevant information from the received signal itself [1]. The Woo filter [2], which uses a phase-coded signal derived from linear FM [3], is ideal for this purpose. While most waveforms used for pulse compression generate random, noise-like sidelobe patterns, which make them hardly practical for sidelobe cancellation, the uniform sidelobe patterns of the Woo filter are promising for such a scheme.

Theoretical derivation: As a symmetric form of the P3 code [3], let a polyphase code s-P3 be defined as

$$s\overline{P3} = \sum_{q=0}^{N} \exp\left[j\frac{\pi}{N}(N-q+1)^2\right] \qquad (1)$$

The Woo filter corresponding to the s-P3 code [2] generates a sidelobe pattern that can be approximated by a unit amplitude curve, which is described as

$$\overline{\chi(q)} \cong \begin{cases} (-1)^N \exp\left[-\frac{\pi}{N}q^2\right] & 1 < q < N-1 \\ -(-1)^N \exp\left[-\frac{\pi}{N}(2N+1-q)^2\right] & N+2 < q < 2N \end{cases} \qquad (2)$$

For a signal of code length $N + 1$, the corresponding Woo filter has N code elements and the mainlobe position at the pulse compression output is designated as $q = (N, N + 1)$. It should be noted that (<) is used instead of (≤) indicating that the sidelobe formulation in eqn. 2 is an approximate form derived on the assumption of a sufficiently large N, and may not be valid for some q values near the ends of the sidelobe pattern. The range sidelobes produced by the Woo filter are defined by a simple transformation of the incoming signal, and so it should be possible to remove unwanted sidelobe energy by a simple manipulation of the received signal.