HIGH SPEED LOW POWER ARCHITECTURE FOR MEMORY MANAGEMENT IN A VITERBI DECODER

Emmanuel BOUTILLON, Nicolas DEMASSIEUX
Telecom Paris , E.N.S.T, 46 rue Barrault, 75634 PARIS CEDEX 13, FRANCE
e-mail : BOUTILLON, DEMASSIEUX@ELEC.ENST.FR

***ABSTRACT*- The management of the surviving-path memory in the Viterbi algorithm is generally performed by Trace-Back or Exchange Register. It has been shown that combining these two techniques leads to efficient realisation [5-7]. In the present work, formal expressions of computational power, memory and latency are presented for several classes of algorithms. For a ν=4, L=64 Viterbi decoder, this formalism helps to find two algorithms that respectively reduce by a factor of 4 and 7 respectively, the computational power compared to a direct Exchange Register. Implementaion results —place&route netlist generated through VHDL synthesis— and theoritical results are in concordance.**

I. INTRODUCTION

Management of the Surviving-Path Memory (SPM) in the Viterbi algorithm is an important problem for VLSI implementation. The well known techniques, Exchange Register (ER) and Trace-Back (TB) present serious limitations [1-4]. ER is a direct implementation whose critical path consists of one multiplexer and one latch, thus allowing high data frequencies. However, it rapidly becomes critical in area and power dissipation when the code size increases. TB, based on RAMs, achieves better integration density but leads to low throughput and high latency. In [5], we proposed a generalised precompilation scheme for surviving-path memory management that overcomes the previously mentioned drawbacks by combining the two techniques. In this paper, the formal expression of the processing power, memory and latency as a function of the algorithm parameters is presented. Using this formalism, two low power VLSI realisations of SPM unit were elaborated for a 16-state (ν=4) Viterbi decoder with a convergence length of L=64.

Section II describes the likeness between ER and TB.. Section III presents the principle of pre-compiling trace-back and section IV gives the formal expression of the architecture parameters for three classes of algorithms using pre-compilation. In the last section, VLSI implementations are presented.

II. TRACE BACK AND EXCHANGE REGISTER ALGORITHMS

Consider a Viterbi decoder with a length constraint k=ν+1 and a convergence length L. Let V_t, a $N=2^\nu$-bit vector, be the decision vector computed by the Add, Compare and Select unit (ACS) at time t, and let "decoding cycle" be the generation time for this vector. Now consider the classical trace-back algorithm. A trace-back on a length of L+H, where H is the extra length of the surviving path traced back after reaching the convergence, allows the decoding of H+1 bits. The constraint flow implies that the computational power P of a TB unit —the number P of trace-back steps that it can perform at each decoding cycle— is equal to the average number of trace-back steps C_{tb} needed to decode one bit:

$$P = C_{tb} = \frac{L+H}{1+H} \text{ TB/Cycle} \tag{1.a}$$

Equation (1.a) shows that the constraint flow can be achieved by several trade-offs between the computational power P and the length of the trace-back L+H. For example, if P = L, then H = 0; if P = 2 then H = L-2.

The minimum number M_{tb} of decision vectors V_t to store depends on two constraints. The first one is static: the memory contains at least L+H vectors. The second one is dynamic: during the trace-back of the surviving path, other decision vectors computed by the ACS unit need to be stored. The processing power P of a TB unit is expressed more explicitly by $P=u_{tb}.v_{tb}$, where u_{tb} stands for the number of trace-back units concurrently working and v_{tb} is the number of trace-back steps executed by each unit in one decoding cycle, i.e. the speed of the trace-back. Thus, the first data of the surviving path is decoded L/v_{tb} cycles after the starting time. The size of the memory needs to be increase by L/v_{tb} N-bit vectors to store the vectors generated during these L/v_{tb} cycles:

$$M_{tb} = L + H + \frac{L}{v_{tb}} \text{ N-bit vectors} \tag{1.b}$$

The latency L_{tb} is the maximum number of cycles required for decoding one bit, i.e. the last bit of a trace back of length L+H at a speed of v_{tb} trace-back steps each cycle.

$$L_{tb} = L + H + \frac{L+H}{v_{tb}} \text{ cycles} \tag{1.c}$$

For the exchange register, the determination of these parameters is straightforward :

$$C_{er} = L \text{ ER/cycle} \tag{2.a}$$
$$M_{er} = L \text{ N-bit vectors} \tag{2.b}$$
$$L_{er} = L \text{ Cycles} \tag{2.c}$$

with C_{er} being the average number of columns processed by the Exchange Register per cycle, M_{er} the number of vectors handled and L_{er} the latency, in decoding cycles. One can note that, for P=L, relations (1.a) and (2.a) lead to:

$$\left| \begin{array}{ll} C_{tb} = L \text{ TB/cycle} & (3.a) \\ C_{er} = L \text{ ER/cycle} & (3.b) \end{array} \right.$$

Moreover, a trace-back implementation requires a $N \rightarrow 1$ multiplexer, where $N = 2^v$ is the number of states of the encoder. Nearly, the same amount of hardware is required to implement an ER (N $2 \rightarrow 1$ multiplexers). This allows to compare both architectures with the same cost unit: TB/cycle = ER/cycle. In figure 2, the two algorithms are represented in a 2D space "processing power" versus "latency". Both have limitations: ER defines only one point in this space and TB has poor performances when the processing power increases (twice the latency of ER for P=L).

III. PRECOMPILED ALGORITHMS

To overcome these limitations, different combinations of TB and ER have already been studied [6-7]. The generalisation of these methods is presented in [5]. Let us briefly recall the principle of precompilation. The continuous Exchange Register carried out on the whole surviving path has unacceptable area and power dissipation. The basic idea is then to perform a partial Exchange Register by generating, and then following, portions of the surviving paths of h-bit width over a depth l. This partial Exchange Register scheme is called ER(h,l). Every $t_1 = n.l$ cycles, with n an integer, this ER(h,l) gives the sequence of h bits of the surviving paths between $t_2 + h$ and $t_2 + 1$, with $t_2 = (n-1).l$. If $h \geq v$, the antecedent at time $t_2 = (n-1).l$ of each node of the trellis at time $t_1 = n.l$ is known. The look-up table thus created enables a very fast Precompiled Trace-Back (PTB).

Figures 1 and Tables 1 and 2 give an example of precompilation using an ER(2,4) for a v=2, L=6 Viterbi decoder. Figure 1 shows the first 8 decision vectors represented in a trellis diagram. Table 2 shows, for each cycle the state of a h=4 bit length ER. Each 4 cycles, here at t=4 and t=8, the h=4 first bits of the surviving path are memorised in Table 3: they represent the Pre-Compiled Surviving path (PSP). $PSP(t_1,t_2)$ corresponding to the 2^v surviving paths, at time t_1, between time t_1 and t_2. The last v=2 bits of $PSP(t_1,t_2)_{node\ i}$ give the number of the node of the surviving path of node i at time t_2. To perform a pre-compiled trace-back, a random state is chosen, for example the node 0. $PSP(8,4)_{node\ 0} = 11\underline{11}$, thus at t=4, its surviving path goes through node 3. $PSP(4,0)_{node\ 3} = 01\underline{01}$ at time t=0, its surviving path goes through node 1. This operation of pre-compiled trace-back is represented in bold in table 2 and in grey in the trellis diagram of figure 1.

Using this formalism, we can now modelize and generalize the algorithms proposed in [6-7] and the algorithm of Pre-Compiled Pointer presented in [5].

IV. EXPRESSION OF ARCHITECTURAL PARAMETERS

Pointer algorithm : ER(v, L)

An ER(v,L) provides, every L cycles, the antecedent at time t-L of each node of the trellis. By definition of the convergence length L, all these antecedent nodes should be equal to the node of the surviving path at time t-L. From this state, a classical trace-back can start: since the convergence has been reached, one symbol is decoded at each trace-back step. If u_{pt} ER(v,L) work in parallel, at each sequence of L/u_{pt} cycles, a decoded state is given to start a TB of length L/u_{pt}. In that case, the algorithm parameters are:

$$\left| \begin{array}{ll} C_{pt} = u_{pt}.v \text{ ER/cycle} + 1 \text{ TB/cycle} & (4.a) \\ M_{pt} = u_{pt}.v + L + \frac{L}{u_{pt}} \text{ N-bits vectors} & (4.b) \\ L_{pt} = L + 2.\frac{L}{u_{pt}} \text{ Cycles} & (4.c) \end{array} \right.$$

Precompilation Trace-Back (PTB): ER(h,h)

An ER(h,h) allows to execute the equivalent of h TB each cycle. The formal expression of the algorithm parameters of PTB are obtained as for a TB algorithm, but with two differences:

- h bits of the surviving path are traced-back at each cycle
- the pre-compiled trace-back starts from a state given by the ER(h,h), thus the first h vectors are already treated and only L-h vectors have to be traced-back before reaching convergence. In n_{ptb} pre-compiled trace-back steps, there are $(n_{ptb}+1).h$ decisions traced-back, which correspond to $(n_{ptb} + 1).h - L$ decoded bits. The constraint flow implies:

$$\left| \, n_{ptb} \leq (n_{ptb} + 1).h - L \right. \qquad (5)$$

thus, the optimal number of pre-compiled trace-back steps is expressed by:

$$\left| \, n_{ptb} = \left[\frac{L-h}{h-1} \right]^+ \right. \qquad (6)$$

where $[X]^+$ is the smallest integer greater than or equal to X. The PTB algorithm parameters are:

$$\left| \begin{array}{ll} C_{ptb} = h \text{ ER/cycles} + h \text{ TB/cycles} & (7.a) \\ M_{ptb} = h + ([\frac{n_{ptb}}{h}]^+ + n_{ptb}).h \text{ N-bit vectors} & (7.b) \\ L_{ptb} = (n_{ptb} + 1).h + n_{ptb} \text{ cycles} & (7.c) \end{array} \right.$$

In equation (7.b), the term $[n_{ptb}/h]^+$ indicates the number of packets of h bits generated during the n_{ptb} cycle of the PTB.

Pre-Compiled Pointer (PCP) : ER(v, L/n)

One can notice that for the ER(h,h), before convergence, only the last v columns of the precompiled tables are used during the trace-back, as shown in table 2. It is thus possible to replace the ER(h,h) by an ER(v,h) generating the minimum information required to trace back the surviving path until convergence. Once convergence is reached, there are two possibilities :

a) All the bits are decoded with a classical trace-back. It is the solution used in [7] where an ER(L,L) (a classical ER) is replaced by an ER(v,L) and a TB of length L.

b) The missing h-v bits of the blocks of length h are decoded by a classical trace-back.

The last solution offers the best trade-off between latency and computational power. This method is called Pre-Compiled Pointer (PCP). For h = L/n, where n is an integer, the performances are:

$$C_{pcp} = \frac{n.v}{h}\ \text{ER/cycle} + \frac{n.v + h\text{-}n}{h}\ \text{TB/cycle} \quad (6.a)$$
$$M_{pcp} = v + n.v\ + (n+1).(h\text{-}v)\ \text{N-bit vectors} \quad (6.a)$$
$$L_{pcp} = \text{Max}(L{+}h{+}n\ ;\ L{+}2.(h\text{-}v){+}n\text{-}1\ )\ \text{cycles} \quad (6.c)$$

In figure 2, all the described algorithms are represented in a 2D space "processing power" versus "latency" for a Viterbi decoder, used in [8], with v = 4 and L = 64. It shows that new interesting trade-offs have been found. For instance, the PTB-ER(8,8) and the PCP-ER(4,8) schemes reduce the computational power by a factor of 4 up to 7, while the latency increases by only 25 %.

V. VALIDATION OF THE COST FUNCTION

To validate the theoretical results, both architectures and a classical ER(64,64) have been implemented in a 0.8 µm standard cell technology up to place&route layout. The netlists have been generated through VHDL synthesis. Except the FIFO of the PCP unit (made of latch), all datas are stored with registers.

Figure 3 shows the architecture of the PTB unit for a simple v=2, L=12 Viterbi decoder. Every four cycles, the blocks are shifted and the first one is initialised with the contents of the ER(4,4). Then, the PTB starts from block 0 to block 4 and stops four cycles later. This corresponds to the trace back of 4.4 =16 bits, from which 4 bits are decoded. Figure 4 gives the architecture of the PCP : the pre-compiled pointer is built with the same architecture used for the PTB, except that the blocks are 2-bit wide instead of 4-bit. Every 4 cycles, the 2 decision vectors which are not processed by the ER(2,4) are stored in a FIFO. A combinatorial trace-back, starting from the pointer given by block 3, provides the 2 decoded bits.

Table 3 presents the number of gates and the power dissipation of the three architectures. According to the computational power, the power dissipation is divided by 4 between ER and PTB and by 6 between ER and PCP. Moreover, the maximum frequencies of the two proposed algorithms are similar with the maximum frequency of the exchange register.

CONCLUSION

The equivalence between TB/cycle and ER/cycle has been established. This equivalence has allowed to determine the formal expressions of computational cost, memory and latency for several surviving path memory algorithms. Applied to a real case, a v=4, L=64 Viterbi decoder, this formal expression leads to new interesting trade-offs between computational power and latency. VLSI realisation (up to the place and route layout) has proved the validity of the method. Compared to a classical exchange register, the two proposed algorithms reduce power dissipation from 19 mW/MHz (ER) to 5 mw/MHz (PTB) and 3 mW/MHz (PCP) with a number of gates area increases of 50 %. These results are very interesting since power dissipation has become one of the most important issues in VLSI circuits.

FIGURES

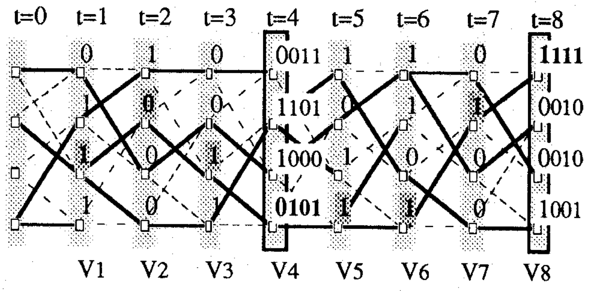


Figure 1: Representation of the decision vectors between t=1 and t=8. The surviving path of all nodes are represented. For t=4 and t=8, the first four bits of the surviving path of the node is represented. It corresponds to the information given by an ER(4,4). A pre-compiled trace-back starting at t=8 from node 0 is indicated (grey and bold).

Nodes	t=1	t=2	t = 3	**t = 4**	t = 5	t = 6	t = 7	**t = 8**
00	0xxx	11xx	011x	**0011**	1xxx	10xx	010x	**1111**
01	1xxx	01xx	000x	**1101**	0xxx	11xx	111x	**0010**
10	1xxx	00xx	101x	**1000**	1xxx	01xx	010x	**0010**
11	1xxx	01xx	101x	**0101**	1xxx	11xx	001x	**1001**

Table 1 : State of an ER of length h=4. The x are computed information but not used.

Nodes	PSP(8,4)	PSP(4,0)
00 = "0"	**11$\underline{11}$ -> "3"**	00$\underline{11}$ -> "3"
01 = "1"	00$\underline{10}$ -> "2"	11$\underline{01}$ -> "1"
10 = "2"	00$\underline{10}$ -> "2"	10$\underline{00}$ -> "0"
11 = "3"	10$\underline{01}$ -> "1"	**01$\underline{01}$ -> "1"**

Table 2 : Pre-compiled Surviving Path (PSP) at time t=8 and t=4

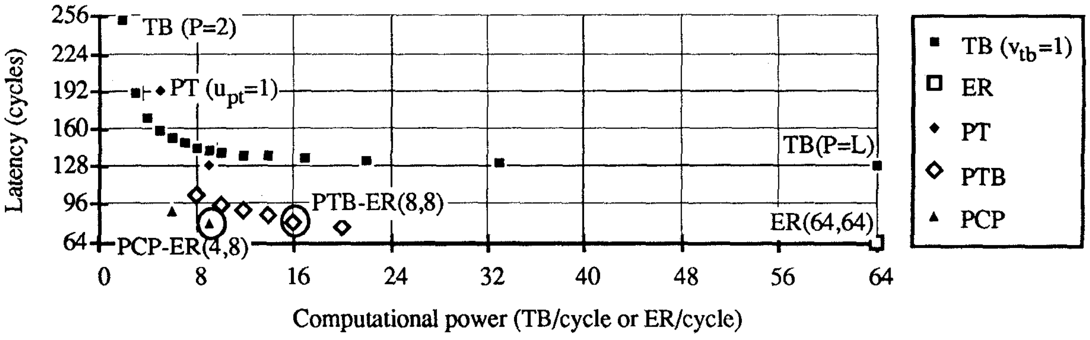
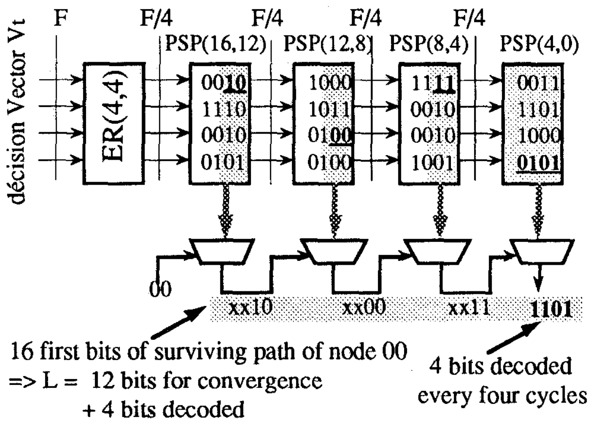
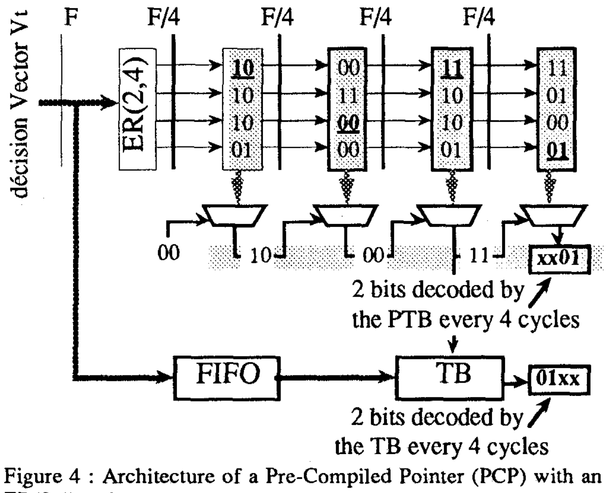


Figure 2: Pepresentation of the different algorithms in the space "computational power" vs "latency"



décision Vector Vt
F
F/4
PSP(16,12)
PSP(12,8)
PSP(8,4)
PSP(4,0)
ER(4,4)
16 first bits of surviving path of node 00
=> L = 12 bits for convergence
+ 4 bits decoded
4 bits decoded every four cycles

Figure 3 : Architecture of a Pre-compiled Trace-Back (PTB) with an ER(4,4) unit (PSP(8,4) and PSP(4,0) are related to figure 1).



décision Vector Vt
F
F/4
ER(2,4)
2 bits decoded by the PTB every 4 cycles
FIFO
TB
2 bits decoded by the TB every 4 cycles

Figure 4 : Architecture of a Pre-Compiled Pointer (PCP) with an ER(2,4) unit.

	ER	PTB	PCP
Processing Power	64	16	8,5
Latency	64	80	80
Number of gates	8752	13670	11000
Frequency	112 MHz	92 MHz	90 MHz
Power dissipation	19 mW/MHz	5 mW/MHz	3 mW/MHz

Table 3: Place and route results of a surviving path memory management unit (v=4, L=64) for ER, PTB and PCP algorithms.

REFERENCES

[1] O. Collins, F. Pollara, "Memory management in traceback Viterbi decoders", DA Progress Report 42-99, Jet Propulsion Lab., Pasadena, CA, (Nov. 1989).

[2] R. Cypher and C. B. Shung, "Generalized traceback techniques for survivor memory management in the Viterbi algorithm,", in Globecom, pp 1318- 1322, (Dec. 1990).

[3] G. Feygin, P. G. Gulak and F. Pollar, "Survivor sequence memory management in Viterbi decoder,", in Third IBM Workshop on ECC, pp 72-90, San Jose, CA, (Sept. 1989).

[4] G. Fettweis, "Algebraic survivor memory management for the Viterbi detectors", conf. proc. of IEEE Int. Conf. on Com., ICC'92, Chicago, paper n°313.4.(June 92)

[5] E. Boutillon, N. Demassieux, "A generalized precompiling scheme for surviving path memory management in Viterbi decoders", in Proc. of ISCAS 93, Chicago, pp 1575-1578, (May 93).

[6] E. Paaske, S. Pedersen, J. Sparso, "An area-efficient path memory structure for VLSI implementation of high speed Viterbi decoders", INTEGRATION, The VLSI journal 12, pp. 79-91, (1991).

[7] J.F. Hellard, "Novel high bit rate viterbi decoder for monodimensional treillis coded modulation on fading channels", Electronics letters, n° 18, (Sept. 93).

[8] C. Berrou, P. Adde, E. Angui, F. Faudeil, "A low complexity Soft Output Viterbi Decoder Architecture", conf. proceeding of IEEE Int. Conf. on Communications, ICC'93, Geneva (May 93).