REALIZATION OF A MONOCHIP CONVOLUTIONAL CODER - VITERBI DECODER, IN ASIC TECHNOLOGY

Philippe Sadot

ALCATEL THOMSON FAISCEAUX HERTZIENS
55, rue Greffühle, 92301 - LEVALLOIS-PERRET

RÉSUMÉ :

Le code convolutionnel de longueur de contrainte 7 et de taux 1/2 (polynômes générateurs 133 et 171) associé au décodage par algorithme de Viterbi, a fait l'objet de normes de la part de la NASA et d'organisations internationales comme INTELSAT et EUTELSAT. Les débits utiles requis atteignent 15 Mbit/s pour certaines applications.

L'objet de cet article est de décrire l'implantation d'un circuit ASIC codeur convolutionnel - décodeur de Viterbi, à haut débit, pour ce code et ses dérivés de taux 3/4 et de taux faibles 1/4 et 1/8.

SUMMARY :

The widely used convolutional code of rate 1/2 and constraint length 7 (generator polynomials 133 and 171), in association with the Viterbi decoding algorithm, has been normalized by several organizations such as INTELSAT, EUTELSAT and NASA. The data rates requirements reach several Mbps (up to 15 Mbps).

This paper describes an ASIC realization of a convolutional coder - Viterbi decoder for this code and the codes derived from it of rate 3/4 and of low rates 1/4 and 1/8.

Introduction

Once very few employed, convolutional codes have been very attracting since A. J. Viterbi invented his now famous algorithm [1,2], which provides the optimal solution to the problem of their decoding. The performance they reach thanks to this algorithm on the usual additive, white, gaussian noise channel (AWGN), better than these of all other codes of same coding rate and equivalent complexity, and also their easy synchronization make that, in every application where the improvement of the receiver's sensibility is looked for, their use is considered and often withheld. International organizations such as INTELSAT, EUTELSAT and the Consultative Committee for Space Data Systems (CCSDS) have normalized these codes for satellite communication or telemetry links. The data rates requested by these organizations reach 15 Mbps for some applications. The universally considered code is the constraint length 7, rate 1/2 code with generator polynomials 133 and 171 (octal notation). New applications appear today, which require a 3/4 rate code, constructed from the 1/2 one by puncturing, i. e. by deletion of some parity symbols after coding.
The major drawback of the Viterbi algorithm is the exponential dependance of its complexity versus the constraint length of the code, which limits its use to the decoding of short codes : the normalized code has length 7 for it has been shown that this length provides the best trade off between complexity and efficiency. Such a code requires the calculation of 64 metrics for the decoding of each data bit, every calculation consisting of two additions, one comparison and one selection ; these 64 operations may be done serially in a unique module, but for high speed operation it is necessary to implement a parallel structure of 64 modules, which leads to a large amount of components and to cost inefficiency. A full custom VLSI realization is possible but also onerous. ASIC technology is nowadays a third way to implement the parallel high speed Viterbi algorithm, which cumulates the advantages of the VLSI realization and of cost effectivness. The matrix sizes now available by some founders and the switching speed obtained in the CMOS 1.5 micron technology make possible to implement this algorithm at high data rates. We have chosen this solution ; our founder is LSI Logic SA.

External specifications

The circuit developped now contains a coder and a decoder whose characteristics are the following :

Technology CMOS 1.5 μm

Coder
- constraint length 7,
- generator polynomials 133 and 171,
- variable coding rate by puncturing for the rate 3/4 and by re-use of the same polynomials for the rates 1/4 and 1/8,

Decoder
- 3-bit quantization soft decision,
- programmable truncation length up to 64 branches,
- integrated fast synchronization device,
- maximum data rate 34 Mbps (25 C, 5V),

- unique power supply 5V, maximum consumption < 3W

Let us recall that the coding gain of the 1/2 code at BER = 10^{-5}, in 3-bit quantization soft decision, is 5.2 dB.

Encoder realization

All the codes listed above are generated from the code of constraint length 7, generator polynomials 133 and 171, whose realization is very easy by use of a six stage shift register whose taps are connected following the coefficients of the generator polynomials.(see figure 1).
The rate 3/4 code is obtained from the 1/2 one by deletion of some parity check symbols according to the deletion pattern 110,101, which transforms the $\{P_n\}$ and $\{Q_n\}$ sequences into the punctured ones
$\{R_n\}=(\ldots,P_n,P_{n+1},P_{n+3},P_{n+4},\ldots)$,
$\{S_n\}=(\ldots,Q_n,Q_{n+2},Q_{n+3},Q_{n+5},\ldots)$.
Let us recall that the contents of the 6-stage shift register is usually referred to as the state of the coder, which can also be viewed as a finite state machine. This representation leads to a state transition diagram called trellis, whose general motif is given on figure 2.

Decoder realization

The decoder comprises essentially five independent functions :

- a de-puncterer (for the rate 3/4), that restores the original rate 1/2 codewords by inserting bits at the places where they have been deleted (these inserted bits will not be taken into account in the further decoding operations),
- a path metric calculator, which extends all the paths of one branch at each incoming codeword,
- a path storage RAM, in which the paths are stored for three truncation lengths, and which is read backwards (traceback algorithm) to find the good path among all the stored paths,
- a fast synchronization device, whose criterium is based on the observation of the metrics' increase speed,
- a timer, which generates from the incoming clock all triggering signals needed by the circuitry.

The de-puncturing device is exactly symmetric to the puncturing device represented on figure 1.

The path metric calculator is a complex function which comprises two subfunctions :

- <u>a branch metric calculator</u>, which determines the distances of the incoming code word (one, two, for or eight bits) to the four possible codewords,
- <u>the path metric computer</u> itself, composed of 32 identical modules, called ACS modules for Addition-Comparison-Selection, each module processing one motif such the one drawed on figure 2. Each computation consists of the comparison of the metrics of the two possible paths reaching every node at time n + 1, and choosing the one which presents the smallest metric, i. e.,

$$M_{2i,n+1} = \mathrm{MIN}(\ M_{i,n} + d_0\ ,\ M_{i+32,n} + d_1\ )$$
and
$$M_{2i+1,n+1} = \mathrm{MIN}(\ M_{i,n} + d_1\ ,\ M_{i+32,n} + d_0\ ),$$
where d_0 and d_1 represent the branch metrics corresponding to this particular ACS module.
The path metrics are positive quantities that do not cease to increase during decoding : as they are stored in a one byte, it is obvious that they have to be reframed sometimes unless they will sature to the all-one byte (FFH), and the decoding operation will fail. Fortunately, it can be shown [2] that the gap between the minimum and the maximum value of the metrics at any time can be upperbounded by $\mu(K-1)$ where μ is the max value of the branch metric and K is the constraint length. It also suffices that the available metric dynamic be greater than this gap. In the case of the 1/2 code, with a 3-bit quantization at the receiving end, this bound is equal to 14*6=84 ; for the 1/4 code with 3 bit quantization, it is equal to 28*6=168 ; for the 1/8 code with 2 bit quantization, 24*6=144. This shows that an 8-bit dynamic is sufficient for the path metric storage. When all the path metrics are greater than 32, one adds 224 to every of them (modulus 256), so that the least of them be equal to 32+224=0 [256] (framing operation).
The 64 path extensions are done in one cycle time by 32 ACS modules.
The outputs of these modules are the 64 decisions taken at each node : if the survivor path is issued from state i, one stores 0 ; one stores 1 if it comes from state i+32. These bits correspond to the data bits to be introduced in the coder to generate the same node transition. These decisions are grouped in a 64-bit word.

The path storage operation consists in putting at each time interval this 64-bit word in a RAM. The decision is strictly optimal if one waits until the end of the message, choose the miminum distance path and also decides the whole message after having received its last bit. This is of course impossible, in particular for continuous stream communications which are almost endless. In fact, we can note, along the construction of the survivor paths, looking the treillis backwards, that these ones converge rather quickly toward a unique path, for which the decisions can be taken without degrading the optimality. The length of the diverging part of the paths is usually referred to as the **truncation length** [3].
The algorithm used here for taking the decisions is called the "Traceback algorithm". It needs a memory organized in three blocks, each of them containing a number of 64 bit words equal to the truncation length (see fig 4). The algorithm works as following : while block 3 is being written straightforward, blocks 2 and 1 are read backwards, twice as fast. An arbitrary path is traced back in block 2, up to the convergence point of all paths ; then the trace back goes on in block 1, where the traced path is the searched one since all the paths should have converged at the end of block 2. Then block 1 is written, blocks 3 and 2 are read, and so on. The corresponding bits are the decoded data bits, but since they are read in reverse sense, they have to be put in a buffer, which will be read backwards in order to regenerate the data bits initial order. The truncation length is a function of the coding rate and of the constraint length of the code. Its value can be optimized by simulations, it is usually set to 32 for the codes of rate 1/2, 1/4 and 1/8, and to 64 for the rate 3/4 code. The RAM used is a three-port RAM, each port of 64 bits, able of achieving in one cycle the write and the two read operations.

The synchronization is obtained by measuring the speed the minimum path metric increase : if it exceeds a threshold obtained by simulations, a synchronisation order is generated. One can also show that this speed is related to the parameter Eb/No of the channel through a one-to-one mapping, so that it is easily possible to give a rather accurate estimation of the quality of the link owing to this property.

Conclusion

The design of this chip is being worked out now at LSI Logic on their own softwares; the estimation of the gates count gives about 65000 useful gates, and the chip size should be close to 15 mm x 15 mm. Of course these values have to be confirmed at the end of the design and of the layout, which should happen at the beginning of 1989. The prototypes are expected current 1989.

References

[1] Viterbi, A. J., 1971, "Convolutional codes and their performance in communication systems", IEEE Trans Com, Vol. COM-19,pp. 751-772.
[2] Viterbi, A. J. & Omura, J. K., 1979, Principles of digital communications and coding, Mc Graw Hill, New-York.
[3] Forney, G. D., Jr, 1973, "The Viterbi algorithm", Proc. IEEE, vol. 61, pp. 268-278

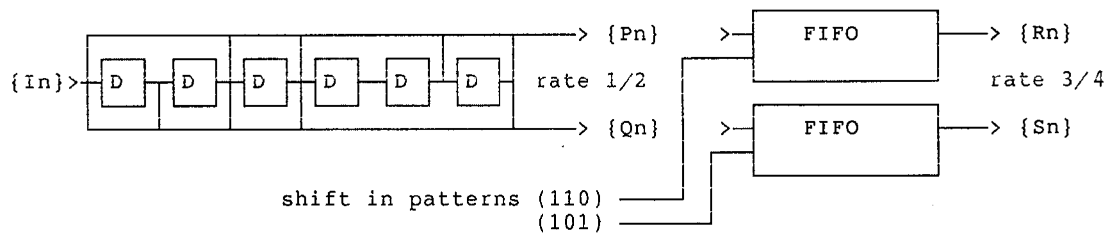
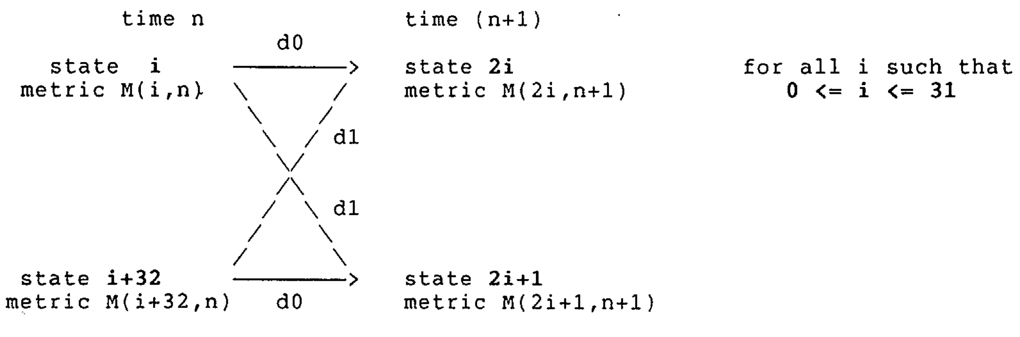
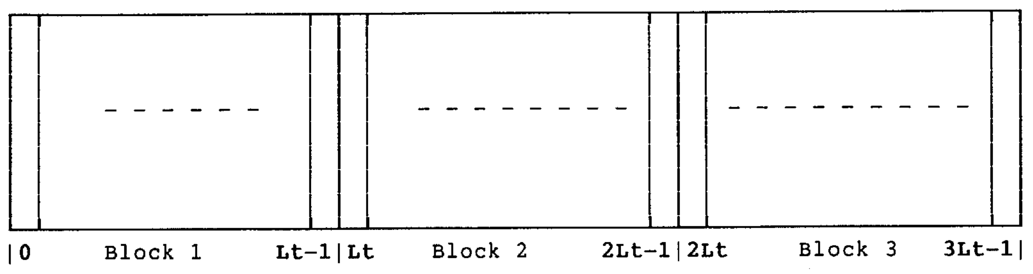


Fig 1 : rate 1/2 and 3/4 convolutional coders



d0, d1 : branch metrics

Fig_2 : Treillis element



A column represents a 64 bit word.

Fig 3 : Organization of the path memory