A COMPARATIVE EVALUATION OF DIRECT AND TRANSFORM VLSI CONVOLVERS ARCHITECTURES

Gilles PRIVAT

CENTRE NATIONAL D'ETUDES DES TELECOMMUNICATIONS
CNET-GRENOBLE, BP 98 38243 MEYLAN CEDEX

This paper addresses the problem of comparing algorithm-specific VLSI architectures implementations of direct vs. transform computation of discrete linear or cyclic convolutions. The choice has been made of *Rectangular* and *Number Theoretic Transforms* involving only straightforward arithmetic, computable without multipliers, and that can be directly mapped onto regular mesh array parallel structures. Approximate logic-complexity as well as asymptotic evaluations are presented that allow to assess, through this illustrative example, the adaptation of classical multiplicative complexity criteria to the VLSI computation medium.

On étudie la comparaison entre des architectures VLSI spécialisées pour la convolution discrète linéaire ou circulaire réalisée par un calcul direct ou au moyen de transformées. On a choisi d'étudier le cas de transformées *rectangulaires* et *en nombres entiers*, qui utilisent une arithmétique simple, sont calculables sans multiplieurs, et peuvent être directement plaquées sur une structure régulière parallèle de tableau à maille carrée. On présente des évaluations approximatives en complexité logique, aussi bien qu'asymptotiques, permettant de mesurer, au travers de cet exemple, l'adaptation des critères classiques de complexité multiplicative à l'implantation des algorithmes en VLSI.

I. INTRODUCTION

The use of Fourier-type transforms having the cyclic convolution property for computing convolutions is common practice for traditional sequential programmable processor implementations [3],[8], resulting in an $O(N^2)$ to $O(N\log N)$ or $O(N)$ computation speedup. Research in this area was given a new impetus through the relatively recent results due to Winograd pertaining to the minimum multiplicative complexity of convolution algorithms which gave, in particular, a theoretical basis to the so-called Rectangular Transforms algorithms, optimum in the rational field [1],[6],[10]. There has also been a continuing interest in Number Theoretic Transforms which attain absolute optimality by embedding the computation into surrogate Galois fields [12].

II. VLSI ALGORITHM EVALUATION

Considering dedicated VLSI architectures implementations changes the perspective radically. In this new setting, classical computational complexity criteria become fully inadapted, forcing one to reconsider anew the whole task of algorithm evaluation.

There does already exist a well-established theoretical body concerning such asymptotic evaluations on the basis of an area-time (AT) or area-time-squared (AT^2) criterion, where die size and computation time are functions of problem size. This *mathematical theory of VLSI* is based on an abstract,— and inevitably crude, model of VLSI computation,

Taking pipelining into account leads to replace the computation latency T by P, the computation period, reciprocal of the throughput rate, which we shall assume is the dominating constraint. For computations such as convolution, FIR filtering, linear transforms which lead to unlooped (unidirectional) structures, full (bit-level) pipeline is always possible and allows to reach a period which is independant of the size of the computing structure, so that an area-only evaluation may be considered to compare between structures having the same input-output bandwith, and the same degree of functional parallelism.

These results should always be taken with caution as constant factors which are concealed in asymptotic evaluations tend to be non-negligible for the problem sizes of practical interest in parallel VLSI architectures, which are bounded by the physical limitations of the silicon medium, rather than by time, as is the case with sequential implementations.

The fundamental limits which asymptotic evaluations allow to capture are on internal memory size, input-output bandwith, and on-chip communication. An illustrative example is given by circuits computing *transitive functions* [4] of size N (that is boolean functions, such as cyclic shift, integer product modulo, cyclic convolution, matrix-vector product, which allow to map any of their input bits onto any of their outputs), where respectively the three lower bounds : $A=\Omega(N)$, $AP=\Omega(N)$ and $AP^2=\Omega(N^2)$ can be obtained.

Those bounds appear infortunately to be of little practical interest as they are at the same time extremely coarse and trivially reachable, by contrast to those which can be derived in the classical sequential algorithm complexity theory.

Attempting, for example, to compare different possible fully parallel VLSI implementations of the Discrete Fourier Transform [5], using an asymptotic evaluation turns out to be rather frustrating, as direct and fast computation are equally optimal from this point of vue, both attaining the $A=\Omega(N^2n^2)$ lower bound, with $P=O(1)$ (full pipeline), N and n being respectively the transform size and word-length.

Such a fully parallel architecture is, however, anything but realistic in present VLSI technologies, and its optimality with regard to the area-time^2 trade-off over either bit-serial or multiplexed architectures is of little more than theoretical interest.

These limitations of asymptotic evaluations in VLSI emphasize the interest of having a mixed approach, retaining traditional logic complexity (transistor-count) evaluation, even on an approximate basis, which obviously suffers, however, from not taking into account connectivity complexity. Regular square mesh arrays allow to minimize this routing area and make a

straightforward logic evaluation nevertheless legitimate (at least as much as an asymptotic one) to base a comparison between two different structures .

This is the reason why an intermediate option has been taken for this study, comparing direct convolution with directly-computed-transform structures, both being implemented as regular nearest-neighbour-connected networks.

III. TRANSFORM COMPUTATION OF CYCLIC CONVOLUTION

Among the numerous transforms allowing "fast" computation of circular convolutions, we shall be interested in those which have the simplest arithmetic. The convolution's N.N Toeplitz circulating matrix **H** undergoes a similarity transformation leading to compute in three stages **Y=HX=BDAX, D** being the diagonal matrix of spectral coefficients. Fourier transform, where **B** and $\mathbf{A}=\mathbf{B}^{-1}=\mathbf{B}^{t}$ are unitary Vandermonde matrices of integral powers of the complex N-th root of unity, can be ruled out as it involves N^2 complex multiplications for both direct and inverse transform phases. Even though this complexity may be classically reduced to $O(N\log N)$, complex arithmetic is by far too cumbersome to be contemplated for application-specific hardware dedicated to real fixed-point convolution.

Number Theoretic Transforms are defined similarly using instead an element of order N in a finite field, with exact and possibly simple arithmetic. Fig. 1 below shows an example of an order 5 Mersenne transform of radix 2 defined in $GF(2^5-1)$.

$$\begin{bmatrix} v0\\ v1\\ v2\\ v3\\ v4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1\\ 1 & 2 & 4 & 8 & 16\\ 1 & 4 & 16 & 2 & 8\\ 1 & 8 & 2 & 16 & 4\\ 1 & 16 & 8 & 4 & 2 \end{bmatrix} \begin{bmatrix} x0\\ x1\\ x2\\ x3\\ x4 \end{bmatrix} \qquad \begin{bmatrix} g0\\ g1\\ g2\\ g3\\ g4 \end{bmatrix} = 25 \begin{bmatrix} 1 & 1 & 1 & 1 & 1\\ 1 & 2 & 4 & 8 & 16\\ 1 & 4 & 16 & 2 & 8\\ 1 & 8 & 2 & 16 & 4\\ 1 & 16 & 8 & 4 & 2 \end{bmatrix} \begin{bmatrix} h0\\ h1\\ h2\\ h3\\ h4 \end{bmatrix}$$

$$u0=g0.v0 \quad u1=g1.v1 \quad u2=g2.v2 \quad u3=g3.v3 \quad u4=g4.v4$$

$$\begin{bmatrix} y0\\ y1\\ y2\\ y3\\ y4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1\\ 1 & 16 & 8 & 4 & 2\\ 1 & 8 & 2 & 16 & 4\\ 1 & 4 & 16 & 2 & 8\\ 1 & 2 & 4 & 8 & 16 \end{bmatrix} \begin{bmatrix} u0\\ u1\\ u2\\ u3\\ u4 \end{bmatrix}$$

Fig.1 Order 5 Mersenne transform

Rectangular Transforms are derived in the rational field (where no proper root of unity exists) following Winograd's theory, by use of a chinese remainder theorem decomposition of the convolution formal polynomial kernel, along the basis of rational cyclotomic polynomials. This ad-hoc derivation is feasible only for small values of N, and can be iterated for larger values by using a Kronecker product of the corresponding transform matrices, following the Agarwal-Cooley multidimensional decomposition and recursive nesting algorithm. The spectral diagonal **D** matrix is M.M, M>N, the direct **A** (N.M) and inverse **B** (M.N) transformation matrices having only simple +1,-1,0 coefficients so that they can be computed without multiplications. Fig. 2 gives an example of an order 4 cyclic convolution computed with 5 spectral multiplications. Fast computation by means of an optimized FFT-type graph is possible, but a general derivation of it is impossible, so that it doesn't lend itself to a regular and systematic implementation (Fig. 3).

$$\begin{bmatrix} y0\\ y1\\ y2\\ y3 \end{bmatrix} = \begin{bmatrix} h0 & h3 & h2 & h1\\ h1 & h0 & h3 & h2\\ h2 & h1 & h0 & h3\\ h3 & h2 & h1 & h0 \end{bmatrix} \begin{bmatrix} x0\\ x1\\ x2\\ x3 \end{bmatrix} \qquad Y = HX$$

$$\begin{bmatrix} d0\\ d1\\ d2\\ d3\\ d4 \end{bmatrix} = 1/4 \begin{bmatrix} 1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\\ 2 & 0 & -2 & 0\\ -2 & 2 & 2 & -2\\ 2 & 2 & -2 & -2 \end{bmatrix} \begin{bmatrix} h0\\ h1\\ h2\\ h3 \end{bmatrix} \qquad d = G\,h$$

$$\begin{bmatrix} y0\\ y1\\ y2\\ y3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 0 & -1\\ 1 & -1 & 1 & 1 & 0\\ 1 & 1 & -1 & 0 & 1\\ 1 & -1 & -1 & -1 & 0 \end{bmatrix} \begin{bmatrix} d0 & 0 & 0 & 0 & 0\\ 0 & d1 & 0 & 0 & 0\\ 0 & 0 & d2 & 0 & 0\\ 0 & 0 & 0 & d3 & 0\\ 0 & 0 & 0 & 0 & d4 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & 0 & -1 & 0\\ 0 & 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} x0\\ x1\\ x2\\ x3 \end{bmatrix}$$

$$Y = B\,D\,A\,X$$

Fig. 2 Order 4*5 Rectangular Transform

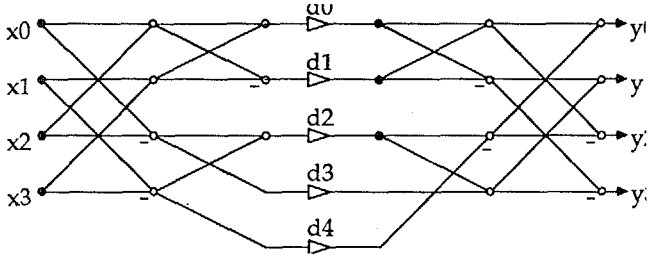

Fig. 3 Rectangular transform computation graph

IV. RECTANGULAR TRANSFORM ARCHITECTURE

The choice made of directly computing the transforms leads to a straightforward RT square mesh array structure, illustrated in Fig. 4. This architecture is a direct fully parallel mapping of the computation and is made up of two symmetric adder-substractor sparse arrays for computing pre- and post-additions (direct and inverse transform) separated by a row of multipliers which compute spectral products. Proper scaling is required between these phases.

If a bit-serial approach is adopted, adder planes are built with simple one cell serial adders with in-place carrying. An evaluation can be made on the basis of these elementary cells which can be supposed to have both inputs latched in a full pipeline scheme. Assuming, for simplicity, constant n bits word width, 2MN such cells are used for adder planes, comprising also dummy delay cells corresponding to the 0 matrix coefficients. Mn similar cells are used for the serial parallel "spectral" multipliers.

This should be compared with a direct parallel computation structure that would require N^2n such cells, as illustrated in Fig. 5. Comparison between the two on this basis (with $n=O(\log N)$) yields a slight advantage in favor of direct computation, growing with convolution order.

Asymptotic evaluations do not make sense in this case as M is no known function of N, and calculated only for tabulated values. Optimal M would be O(N), following Winograd's results, but would lead to non ±1 coefficients, so that it is preferable to choose larger values of M that maintain this property and are stable under tensor product iteration. The problem is then that M grows quite fast with N. It is interesting to note that neither the direct nor the transform computation are optimal on the $AP^2=\Omega(N^2n^2)$ basis in the bit-serial case ($P=O(n)$, $AP^2=O(N^2n^3)$. The direct structure only would be "optimal" in a bit-parallel approach with full pipeline ($P=O(1)$). Such a fully parallel architecture, with bit-level pipeline obtained through input bits skewing (and antisymmetric output bits deskewing) and carry chain latching can also be derived from the rectangular transform structure of Fig.4. It makes poor use of the RT algorithm, compared to the bit-serial structure, as adder arrays can't be (asymptotically) compacted in less area than parallel-parallel multiplier arrays.

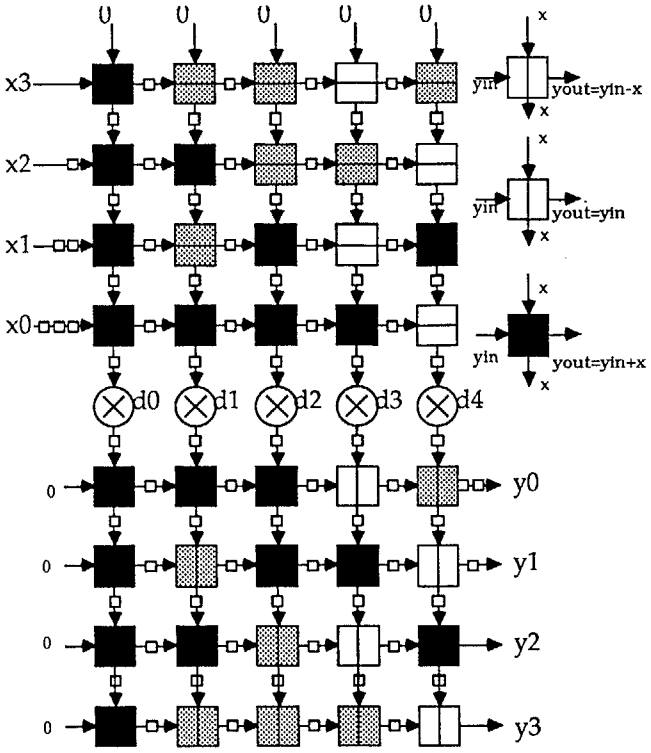

Fig. 4. Rectangular Transform parallel architecture.

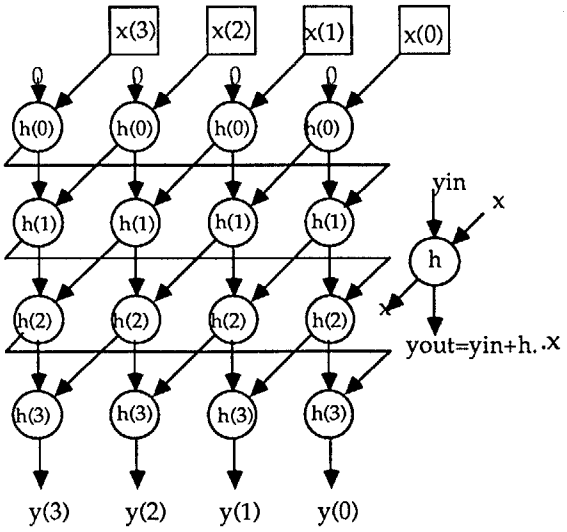

Fig .5. Direct cyclic convolution parallel architecture

Another possibility is to use a multiplexed one multiplier structure that requires M iterations to output its N results in parallel (Fig. 6). This structure relates to the asymptotic bound A=Ω(Nn) (at least that number of memory points is necessary to store a complete input, even if one operates in a bit-serial fashion). It is, with P=O(M), (assuming it is pipelined down to the bit level) far from optimal from the AP^2 point of vue. This architecture should be compared with a similar N multiplier cells direct computation structure that would require N iterations to output N results (Fig. 7). This structure can also be pipelined (inputs being time-skewed and circulating coefficients broadcast to all cells), so as to attain P=O(N). Comparison between the two is interesting, as the RT structure is better in strict area performance ($Nn+n^2$ vs. Nn^2 basic cells, in the bit-parallel case), while it is slightly inferior with an AP figure of merit.

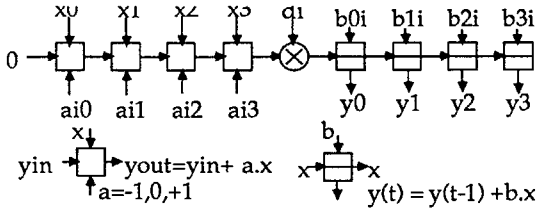

Fig. 6. Rectangular transform multiplexed architecture

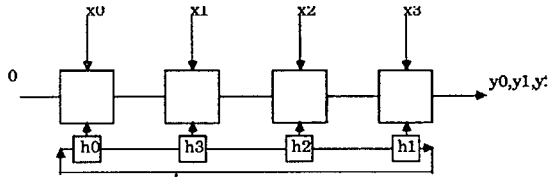

Fig. 7. Direct cyclic convolution architecture

V. NTT ARCHITECTURE

The main problem with rectangular transforms being the uncontrolled growth of the transform dimension, one is naturally leaded to use NTTs for which the number of general (spectral) multiplications is always N. It is known that when the transform radix is ±2 or a power of 2, the direct and inverse transform themselves are computable without multiplications, but only shifts and additions. This nice property is offset, however, by some very troublesome constraints relating transform size and word-length (corresponding to the cardinality of the computation field), which limits very severely the practical use of these transforms.

The surrogate computation field (or ring) must be chosen large enough so that end results do not overflow (intermediate overflow being always possible). Fermat (2^n+1, $n=2^u$, u= 1,2,3,4,) or Mersenne (2^P-1, p= 5,7,13,17,19, 31...) primes are the most common choice for moduli, as they allow simple arithmetic and 2 is a root of unity.

Arithmetic modulo Mersenne numbers reduces to classical one's complement arithmetic, and transforms of length p and 2p can be defined which have respectively radix 2 and -2. They were chosen as an example for this study, as their main drawback (non-composite transform-length, precluding the use of a Cooley-Tuckey computation scheme), was of no consequence, having regard to the option of direct computation taken. A number of devices are available to overcome the constraint of rigid relationship between transform order and word-length [1], [2].

A bit-serial or multiplexed architecture similar to those presented for RT would be relatively cumbersome to implement using one's complement arithmetic, requiring complex control for feedback of the MSB carry output into the LSB carry input. Products by 2^i in the direct and inverse transform stages are also realized with *circular* shifts. A fully parallel scheme allows to map these in the intercell routing pattern, leading to the structure presented in Fig. 8 Exact arithmetic means constant word-length and no overflow which result, however in an overhead rather than a gain, compared to a classical "open arithmetic" scheme with progressive scaling and MSB extension. End-around carry feedback precludes the use of a digit-skewed pipeline scheme, forcing to revert to a carry-save scheme, which is, in this case, less efficient, requiring 3 complementary half-adder stages for final carry propagation. With full pipeline, the structure of Fig. 8 comprises $4Nn^2+2N^2n$ full adder cells. A comparison on this basis with a direct fully parallel structure gives a slight advantage in favor of the NTT structure, growing with transform order.

This result is misleading since its asymptotic performance is in fact dominated by the routing area, in accordance with the $A=\Omega(N^2n^2)$ lower bound.

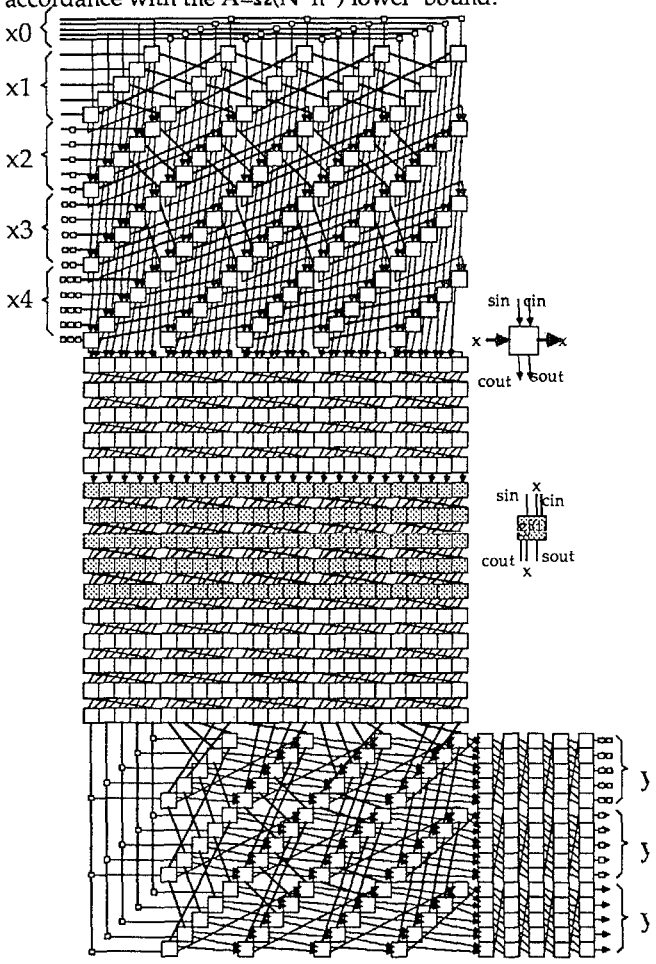

Fig. 8 Mersenne Transform Architecture

VI. LINEAR CONVOLUTION ARCHITECTURE [13]

Circular convolutions are of little practical interest for themselves outside their use for computing Fourier or cosine transforms [11], [12], and first of all, linear convolutions. An N-1 terms overlapping is always necessary to compute by length P blocks a convolution having an N coefficients kernel through the use of a circular convolution of order Q=N+P-1. The classical overlap-save scheme (Fig.9) will generally be preferred, as it doesn't involve extra additions. The cyclic convolution structures presented before may be simplified in this case as only P outputs are needed each time one computes a length Q circular convolution. It should be noted that there exists, at least for rectangular transforms, an optimum value of P for each N, for which the ratio of transform dimension to block-length is minimum [1].

Any (slight) advantage that could exist in favor of transform structure on the basis of a cyclic convolution comparison is in fact lost when one turns to linear convolutions. This is mainly due to the necessity of breaking up the convolution into overlapping sections, while it is "naturally" computed in a pipeline fashion on a continuous stream of data. Such block computation could be justified only in the unlikely case where one would need a throughput higher than what can be obtained with a one bit-parallel sample per bit-level pipeline iteration , or rather if one wanted to trade a lower operating frequency against a larger input-output bandwidth. Anyway, a bit-serial or multiplexed structure *with overlap-save computation* would obviously be counterproductive, while a fully parallel structure is unrealistic to consider in present VLSI technologies.

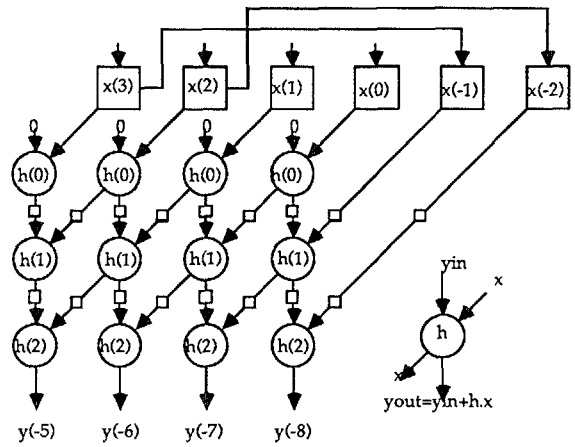

Fig. 9. Overlap-save parallel structure for computation of a linear convolution of length 3 by 4 terms sections

VII. CONCLUSION

Rough as they are, these evaluations allow to gain insight into the trade-offs at work for the choice of a specialized VLSI architecture under throughput and area constraints. It may seem surprising, or disappointing, that, in general, no advantage can be gained from the use of those clever "fast" algorithms which have been the subject of such extensive investigation in the classical signal processing algorithm design field. But their greater conceptual complexity being in conflict with the new pervading VLSI constraints of regularity, modularity and spatial locality, a choice in favor of them would be justified only if a definite advantage could be proved. As we have attempted to demonstrate for the examples chosen, this doesn't seem to be the case.

REFERENCES

[1] J.H. Mc Clellan and C.M. Rader, Eds, *"Number Theory in Digital Signal Processing"*, Prentice-Hall Signal Processing Series, 1979.

[2] H.J. Nussbaumer, *"Fast Fourier Transform and Convolution Algorithms"*, Springer Verlag, Berlin, 1982.

[3] D.F. Elliott and K.R. Rao, *"Fast Transforms"*, Academic Press, 1982.

[4] J. Vuillemin, "A Combinatorial Limit to The Computing Power of VLSI Circuits", *IEEE Trans. Comput.*, vol. C-32, n° 3, Mar. 83.

[5] C.D. Thompson, "Fourier Transforms in VLSI", *IEEE Trans. Comput.*, vol. C-32, n°11, Nov. 1983.

[6] L. Auslander, J. Cooley, A. Silberger, "Numerical Stability of Fast Convolution Algorithms for Digital Filtering", in *VLSI Signal Processing*, R. Cappello *et al.*, Eds., IEEE Press, 1984.

[7] M. Bhattacharya and R.C. Agarwal, "Number Theoretic Techniques for Computation of Digital Convolution", *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, n°3, June 1984

[8] R.E. Blahut, *"Fast Algorithms for Digital Signal Processing"*, Addison Wesley, 1985.

[9] P.R. Cappello, and K. Steiglitz, "A Note on "Free Accumulation" in VLSI Filter Architectures", *IEEE Trans. Circuits and Systems*, vol. CAS-32, n° 3, Mar. 1985.

[10] J.K. Pitas and A.N. Venetsanopoulos, "Two-dimensional Realization of Digital Filters by Transform Decomposition", *IEEE Trans. Circuits and Systems*, vol. CAS-32, n° 10, Oct. 85.

[11] R.M. Owens and J. Ja'Ja', "A VLSI Chip for the Winograd/Prime Factor Algorithm to Compute The Discrete Fourier Transform", *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, n°4, Aug. 86.

[12] P. Duhamel, *"Algorithmes de Transformées Discrètes Rapides pour Convolutions Cycliques et de Convolution Cyclique pour Transformée Rapide"*, Thèse d'Etat, Université de Paris-Sud, Septembre1986.

[13] G. Privat, *"Architectures Spécialisées de Circuits VLSI pour le Traitement Numérique du Signal"*, Thèse pour le Doctorat de l'ENST, Paris, Décembre 1986.