



Determination of a specialized processor for the resolution of linear systems arising in the application of the finite element method

Emmanuel Casseau, Dominique Degrugillier

► To cite this version:

Emmanuel Casseau, Dominique Degrugillier. Determination of a specialized processor for the resolution of linear systems arising in the application of the finite element method. *Journal de Physique III*, 1993, 3 (3), pp.531-538. 10.1051/jp3:1993146 . jpa-00248938

HAL Id: jpa-00248938

<https://hal.science/jpa-00248938>

Submitted on 4 Feb 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Classification
Physics Abstracts
41.90 — 02.60

Determination of a specialized processor for the resolution of linear systems arising in the application of the finite element method

Emmanuel Casseau and Dominique Degrugillier

Ecole Nationale Supérieure des Télécommunications de Bretagne, Département Electronique-Physique, Laboratoire Circuits Intégrés Télécom, B.P. 832, Brest Cedex, France

(Received 17 March 1992, revised 18 August 1992, accepted 18 September 1992)

Résumé. — Une machine spécialisée pour la résolution de systèmes linéaires du type $[A]\{\mathbf{x}\} = \{\mathbf{b}\}$ par la méthode de décomposition LU est décrite. Cette machine utilise des circuits spécifiques VLSI connectés suivant une structure maillée du type systolique. Ce principe d'architecture parallèle permet un gain de temps de calcul important et rend ainsi plus exploitables les méthodes numériques qui nécessitent ce type de résolution.

Abstract. — A specialized machine for the resolution of linear systems of $[A]\{\mathbf{x}\} = \{\mathbf{b}\}$ type by the LU decomposition method is presented. That machine uses specific VLSI circuits connected in a mesh structure of a systolic kind. That parallel structure principle allows a significant saving of computation time, so make the numerical methods that require this kind of resolution more efficient.

Introduction.

Numerical methods used in electromagnetic structure studies require large computer resources (computation time, memory capacity and control). The finite element method is a case in point, since in many cases it requires the resolution of a large linear system $[A]\{\mathbf{x}\} = \{\mathbf{b}\}$ where $[A]$ is a known band square matrix of order n with band width $w = p + q - 1$ called stiffness matrix, $\{\mathbf{x}\}$ an unknown vector with n components and $\{\mathbf{b}\}$ the forcing vector. As an example, the study of an electromagnetic structure that propagates a quasi T.E.M. mode like the microstrip line often leads to $n = 2\,000$ and $p = q = 200$.

Even if current computer power, and a better knowledge of the finite element convergence properties have significantly decreased the analysis time, it still remains excessive.

A way to solve this problem is to use a specialized machine, specific VLSI circuits based, which makes use of optimized parallel structures.

In this paper, we present the structure and the results of such a machine for the resolution of the linear system $[A]\{\mathbf{x}\} = \{\mathbf{b}\}$ described above.

1. Resolution methods for the linear system $[A]\{x\} = \{b\}$.

All the methods for obtaining the solution of the linear system $[A]\{x\} = \{b\}$ can be classified as indirect or direct.

The indirect methods or iterative methods are based on the idea of gradually improving the approximate solution beginning with an initial vector $x^{(0)}$. The main feature of these methods is the convergence speed that depends on initialization.

The direct methods are generally based on the principle of a triangular decomposition of the matrix $[A]$. The major drawback of such methods is propagation (and therefore accumulation) of rounding errors that do not always ensure numerical stability.

However, direct method algorithms underscore more significant possibilities of using optimized parallel structures for a specialized machine implementation.

Regarding the finite element method, which is distinguished by a positive definite symmetric matrix $[A]$ so with a « weighty » diagonal [1], using a direct method without pivoting is suitable.

Thus, we choose the LU decomposition direct method, where we decompose the matrix $[A]$ into a two matrix product, $[L]$ lower triangular and $[U]$ upper triangular.

$$[A] = [L][U] = \begin{bmatrix} \triangle & 0 \\ 0 & \triangle \end{bmatrix} \begin{bmatrix} \triangle \\ 0 \end{bmatrix}.$$

Given $[L]$ and $[U]$, we can derive $\{z\}$ from $\{b\}$ by a forward substitution process to solve $[L]\{z\} = \{b\}$ then derive $\{x\}$ from $\{z\}$ by a back substitution process to solve $[U]\{x\} = \{z\}$.

This direct method of decomposition allows the use of the frontal method [2, 3]. We can perform together the assembling of the stiffness matrix and the LU decomposition that speeds up the finite element method computation time and significantly reduces the memory requirements.

Therefore, for the resolution of the finite element method linear system, we can make use of two degrees of parallelism :

— the first one with the assembling of $[A]$ and the linear system resolution as we have just seen ;

— the second one with the resolution of the linear system, using an optimized parallel structure, presented here, which is able to compute quickly the solution of that system.

2. Parallel structure for the resolution of the linear system $[A]\{x\} = \{b\}$.

2.1 IMPLEMENTATION OF PARALLELISM IN THE LU DECOMPOSITION. — The value of the useful coefficients of triangular matrices $[L]$ and $[U]$ can be computed according to the following recurrences :

equations	computation domain	
$a_{ij}^1 = a_{ij}$	$1 \leq i \leq n, i - q + 1 \leq j \leq i + q - 1$	with $1 \leq j \leq n$

$u_{kj} = a_{kj}^k$	$k \leq j \leq k + q - 1$	with $1 \leq k \leq n, j \leq n$
---------------------	---------------------------	----------------------------------

$\ell_{ik} = a_{ik}^k / u_{kk}$	$k + 1 \leq i \leq k + p - 1$	with $1 \leq k \leq n - 1, i \leq n$
---------------------------------	-------------------------------	--------------------------------------

$a_{ij}^{k+1} = a_{ij}^k + \ell_{ik}(-u_{kj})$	$k + 1 \leq i \leq k + p - 1, k + 1 \leq j \leq k + q - 1$	
	with $1 \leq k \leq n - 1, i \leq n, j \leq n$	(1.4)

$a_{ij}^{k+1} = a_{ij}^k$	$k + p - 1 \leq i \leq n, i - q + 1 \leq j \leq i + q - 1$	
	with $1 \leq k \leq n - 1, 1 \leq j \leq n$	(1.5)

For every equation, we specify the computation domain, i.e. the set of values that can have indices i, j, k of the data of the equation.

The value of the coefficients u_{kj} and ℓ_{ik} is determined from equation (1.4) where a_{ij}^{k+1} is computed with a set of accumulations.

For this equation, defined by its computation domain $D = \{(i, j, k), 1 \leq k \leq n-1, k+1 \leq i \leq k+p-1 \text{ with } i \leq n, k+1 \leq j \leq k+q-1 \text{ with } j \leq n\}$, for every coordinate triple (i, j, k) , we get a computation point, where we have to compute the function $a_{out} = a_{in} + \ell_{in} * (-u_{in})$.

Then we observe that several computation points make use of the same coefficients ℓ_{ik} . The same is true for coefficients u_{kj} .

Two possibilities are then conceivable :

- i) either, for every computation point, to call the requisite coefficients ℓ_{ik} and u_{kj} , therefore reading the same data as many times as there are computation points using that data,
- ii) or to make coefficients ℓ_{ik} and u_{kj} circulate from one computation point to another.

For example, ℓ_{ik} will be used by every coefficient a_{ij}^{k+1} where j ranges from $k+1$ to $k+q-1$, and the coefficient ℓ_{ik} of computation point $(i, j, k+1)$ is provided by computation point $(i, j-1, k+1)$.

We formally express solution ii by the following system of recurrent equations :

equations computation domain

input equation :

$$\begin{aligned} A(i, j, k) = a_{ij} \quad & 1 \leq i \leq n, i-p+1 \leq j \leq i+q-1 \quad \text{with } 1 \leq j \leq n \quad (2.1) \\ & k = \max(i-p, j-q) \text{ if } \max(i-p, j-q) > 0 \\ & k = 0 \quad \text{if } \max(i-p, j-q) \leq 0 \end{aligned}$$

computation equations :

$$A(i, j, k) = A(i, j, k-1) / U(i, j, k) \quad 1 \leq k \leq n-1, j = k, k+1 \leq i \leq k+p-1 \quad \text{with } j \leq n \quad (2.2)$$

$$A(i, j, k) = A(i, j, k-1) \quad 1 \leq k \leq n, i = k, k \leq j \leq k+q-1 \quad \text{with } j \leq n \quad (2.3)$$

$$U(i, j, k) = U(i-1, j, k) \quad 1 \leq k \leq n-1, k \leq j \leq k+q-1, k+1 \leq i \leq k+p-1 \quad \text{with } j \leq n, i \leq n \quad (2.4)$$

$$U(i, j, k) = A(k, j, k) \quad 1 \leq k \leq n, i = k, k \leq j \leq k+q-1 \quad \text{with } j \leq n \quad (2.5)$$

$$L(i, j, k) = L(i, j-1, k) \quad 1 \leq k \leq n-1, k+1 \leq i \leq k+p-1, k+1 \leq j \leq k+q-1 \quad \text{with } j \leq n, i \leq n \quad (2.6)$$

$$L(i, j, k) = A(i, k, k) \quad 1 \leq k \leq n-1, j = k, k+1 \leq i \leq k+q-1 \quad \text{with } i \leq n \quad (2.7)$$

$$A(i, j, k) = A(i, j, k-1) + L(i, j, k) * (-U(i, j, k)) \quad 1 \leq k \leq n-1, k+1 \leq i \leq k+p-1, k+1 \leq j \leq k+q-1 \quad \text{with } i \leq n, j \leq n \quad (2.8)$$

output equations :

$$\ell_{ik} = A(i, k, k) \quad 1 \leq k \leq n-1, k+1 \leq i \leq k+p-1 \quad \text{with } i \leq n \quad (2.9)$$

$$u_{kj} = A(k, j, k) \quad 1 \leq k \leq n, k \leq j \leq k+q-1 \quad \text{with } j \leq n \quad (2.10)$$

Then we can consider the use of a parallel structure as an array of elementary cells where the pipeline feature of data (circulation of the data from one computation point to one another) is operated.

Equations (2.1) to (2.10), said to be uniform, properly express the evolution of the data and the computations of such a structure since coefficients ℓ_{ik} and u_{kj} formulated like that circulate in the array while coefficients $A(i, j, k)$ accumulate their value through that same array.

A systematic method, called the dependence mapping method introduced by Quinton [4], allows, from the system of uniform equations, to characterize arrays which are solutions of the given problem where cells communicate in a synchronous pipeline mode. This kind of array is called a systolic array [5].

2.2 TWO DIMENSIONAL SYSTOLIC ARRAY FOR LU DECOMPOSITION. — A systolic solution for LU decomposition is shown in figure 1.

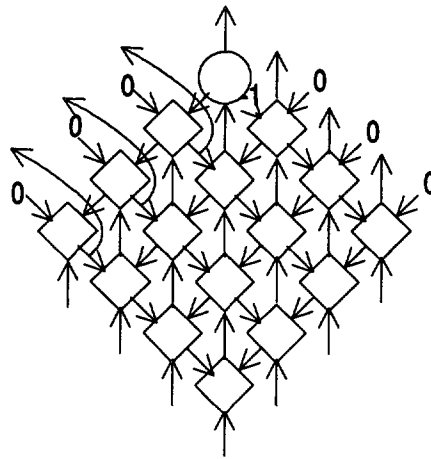


Fig. 1. — Two dimensional systolic array for LU decomposition ($p = q = 4$).

The square elementary cells compute an inner product (multiply and add) to obtain coefficients $A(i, j, k)$ as explained in figure 2a. The cell denoted by a circle performs the division to obtain coefficient ℓ_{ik} .

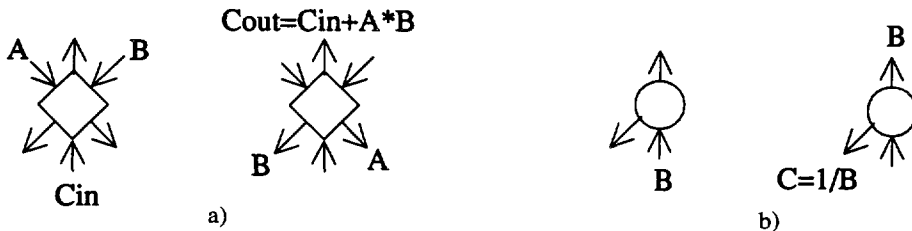


Fig. 2. — a) The inner product processor. b) The division processor.

This systolic array needs a computation time of $(3 \cdot n \cdot T)$ where T is the period of the clock that synchronises data through the systolic machine. This computation time can be compared with time needed to compute the $(2 \cdot n \cdot p \cdot q)$ operations (memory access + computation) for a

sequential algorithm (use of a Von Neuman computer). Therefore we get a significant increase in computation speed.

Another attractive feature of this array is that its size is independent of the order n of the matrix $[A]$, since it requires $(p \cdot q)$ elementary processors.

However, the number of computing cells is high, as is the input-output (I/O) number, $(2 \cdot (p + q - 1))$, which gives rise to a problem of connection since the finite element method requires data to be coded in a scientific representation of at least 32 binary digits.

Therefore, we have to establish a different systolic array which has on the one hand a reasonable number of cells and, on the other hand, not too many inputs-outputs. Furthermore, this structure should still have a rapid computation time.

2.3 ONE DIMENSIONAL SYSTOLIC ARRAY FOR LU DECOMPOSITION. — To overcome the problem of the sizeable number of cells and inputs-outputs of the bidimensional array, due to the 3 indices i, j and k taken simultaneously, the idea is to search for a structure, still of a systolic kind, that makes computations, at a given time, with data indexed by two indices, the third then being intentionally fixed. In this case, the systolic solution is a one dimensional array, or so-called linear array.

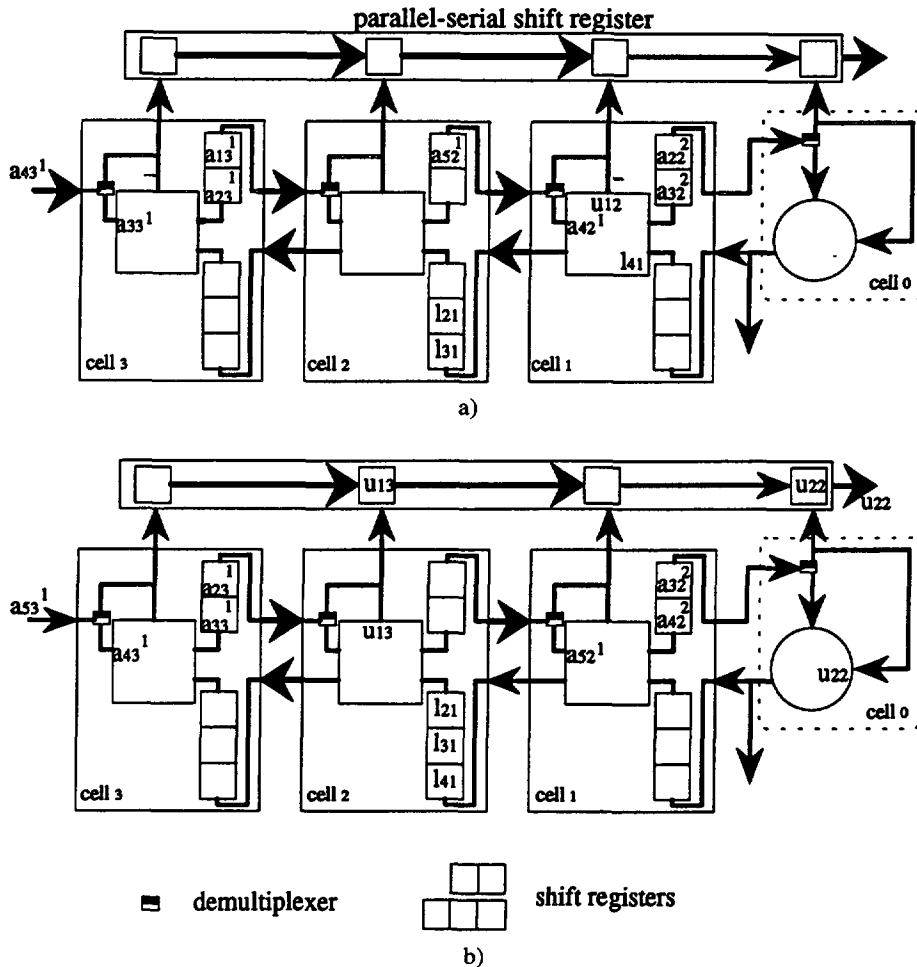


Fig. 3. — One dimensional array for LU decomposition ($p = q = 4$).

For our purpose, we have chosen to fix the index i . A conceivable solution associated with this principle is shown in figure 3 where the written coefficients, according to equations (1.1) to (1.5), correspond to the data into the array at a precise instant. Figures 3a and 3b illustrate two successive snapshots of the array operation.

This systolic array is composed of q elementary cells. The $(q - 1)$ rectangular cells are composed of :

- one processor, denoted by a square, which computes the inner product according to equations (1.4) and (2.8) and according to figure 4 ;
- one first shift register of length $(q - 2)$ which memorizes the results a_{ij}^k from the processor, i.e. coefficients $A(i, j, k)$ for a given j ;
- one second shift register of length $(q - 1)$ which sends the data ℓ_{ik} (i.e. $L(i, j, k)$) to the processor ;
- one demultiplexer that switches data a_{ij}^k either toward the input a_{in} of the processor or toward the parallel-serial shift register and the input u of the processor in order to memorize the coefficient u_{kj} (i.e. $U(i, j, k)$) in this processor.

The array is also composed of a cell represented by a dotted line which includes :

- one processor, denoted by a circle, which computes the division according to equations (1.3) and (2.2) and according to figure 4 ;
- one demultiplexer that switches data a_{ij}^k either toward the input a_{in} of the processor or toward the parallel-serial shift register and the input u of the processor in order to memorize the coefficient u_{kk} (i.e. $U(i, k, k)$) in this processor.

Finally, a parallel-serial shift register of length q allows the serial output of coefficients u_{ij} .

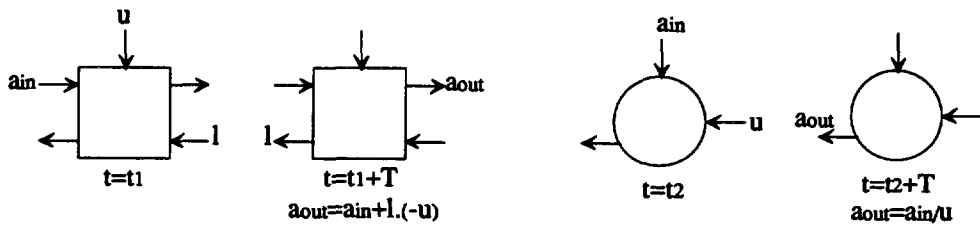


Fig. 4. — Processor operations.

Coefficients of the matrix $[A]$ are distributed to the array input (cell 3 of Fig. 3) in accordance with a *column stream*, which means that we provide each clock cycle a_{11} , then a_{21} , a_{31} , ..., a_{12} , a_{22} , a_{32} ..

Coefficients a_{ij}^k move into the array from the left to the right in accordance with the column stream appearance imposed by the input data stream.

Coefficients ℓ_{ik} move into the array in accordance with a column stream in the direction opposite to that of coefficients a_{ij}^k .

This solution has an efficiency of $1/2$, which means that a cell is active only one clock cycle every two clock cycles. A hardware modification of the elementary cell regarding the synchronous flow of the inner data with multiplexing allows us to obtain an efficiency of 1 while dividing the number of processors by two.

The computation time needed for the LU decomposition is, for this structure, $(2 \cdot n \cdot p \cdot T)$. Therefore we retain a significant computation speed-up as compared with a sequential computer.

Furthermore, as was our purpose, the input-output number is reduced to a minimum since only one input and two outputs (one for coefficients ℓ_{ik} , one for coefficients u_{kj}) are sufficient.

The linear solution presented above therefore constitutes a good speed-size compromise. While being reasonable as regards the size, it allows the computation time of the LU decomposition to be significantly reduced.

2.4 PARALLEL STRUCTURE FOR THE RESOLUTION OF THE LINEAR SYSTEM $[A]\{x\} = \{b\}$. — When the coefficients of the two lower and upper triangular matrices resulting from the LU decomposition of the matrix $[A]$ are computed, we solve the two triangular systems, $[L]\{z\} = \{b\}$ to derive $\{z\}$ and $[U]\{x\} = \{z\}$ to derive $\{x\}$.

The vector $\{g\}$ of a triangular system $[C]\{g\} = \{h\}$ of order n and band width p can be computed by the following recurrences :

$$\begin{array}{ll} \text{equations} & \text{computation domain} \\ g_i^0 = 0 & 1 \leq i \leq p \end{array} \quad (3.1)$$

$$g_i^{i-p} = 0 \quad p < i \leq n \quad (3.2)$$

$$g_i^k = g_i^{k-1} + c_{ik} * g_k \quad 1 \leq i \leq n, \quad i - p + 1 \leq k \leq i - 1 \quad \text{with} \quad 1 \leq k \quad (3.3)$$

$$g_i = (h_i - g_i^{i-1})/c_{ii} \quad 1 \leq i \leq n. \quad (3.4)$$

As previously, the application of the dependence mapping method provides a one dimensional systolic solution.

We can use a single systolic array to solve the two triangular systems $[L]\{z\} = \{b\}$ and $[U]\{x\} = \{z\}$, then we need an array composed of $\lceil \max(p, q)/2 \rceil$ elementary cells. Of these elementary cells, the last one in the array computes the division $g_{out} = (h_{in} - g_{in})/c_{in}$ according to equation (3.4) of the recurrent equation system above. The other cells compute the inner product as required by equation (3.3).

Therefore, the resolution of the linear system $[A]\{x\} = \{b\}$ requires the implementation of two unidimensional systolic arrays; one array for the LU decomposition of the matrix $[A]$ and one array for the resolution of the two triangular systems $[L]\{z\} = \{b\}$ and $[U]\{x\} = \{z\}$. Three computation phases are then necessary for the resolution of the linear system :

- the first phase is for the LU decomposition,
- the second phase is for the resolution of the lower triangular system $[L]\{z\} = \{b\}$ to derive $\{z\}$,

Table 1

	Conventional computer of the Von Neuman kind	Systolic structures based computation unit
Computation cell number	1	$\lceil q/2 \rceil + \lceil \max(p, q)/2 \rceil$
I/O data number	2	2
Computation cycle number	$2 \cdot n(p \cdot q + p + q)$	$2 \cdot n(p + 1)$

— the third and last computation phase is for the resolution of the upper triangular system $[U]\{\mathbf{x}\} = \{\mathbf{z}\}$ to derive $\{\mathbf{x}\}$.

Actually, it is feasible to compute at the same time both the LU decomposition of matrice $[A]$ and the triangular system $[L]\{\mathbf{z}\} = \{\mathbf{b}\}$, thus increasing the speed of the resolution of the linear system $[A]\{\mathbf{x}\} = \{\mathbf{b}\}$.

The features of the whole systolic structure for the resolution of a linear system $[A]\{\mathbf{x}\} = \{\mathbf{b}\}$, defined by n , p and q are given in table I, with the features of a conventional computer of Von Neuman kind as a comparison.

Conclusion.

A behavioral model of the whole systolic structure has been written and simulations have been made. These simulations have enabled operation of the systolic machine and the features presented in table I to be validated.

Therefore, we can now start to tackle the C.A.D. (computer aided design) phase for the design of the systolic machine. This work, now in process, concerns the design of two elementary cells, one for the inner product and one for the division, discussed above.

Once those cells have been implemented, the systolic machine, regarded as a computation unit for the resolution of linear systems of $[A]\{\mathbf{x}\} = \{\mathbf{b}\}$ kind, can be connected to a « host computer » to relieve it of that particular demanding task as far as computation time and memory capacity are concerned, when investigating electromagnetic structures, or others, using the finite element method.

References

- [1] GOLUB G. H., VAN LOAN C. F., *Matrix Computation*, Second Edition (The Johns Hopkins University Press, Baltimore and London).
- [2] ZIENKIEWICZ O. C., *The Finite Element Method* (MacGraw-Hill, 1981).
- [3] IMBERT J. F., *Analyse des structures par éléments finis*, seconde édition (Cepadues, Toulouse, 1984).
- [4] QUINTON P., ROBERT Y., *Algorithmes et Architectures Systoliques*, Etudes et Recherches en Informatique (Masson, 1989).
- [5] KUNG H. T., Why Systolic Architecture ? *Computer* **15** (1982) 1.