PAPIA: PYRAMIDAL ARCHITECTURE FOR PARALLEL IMAGE ANALYSIS

V. Cantoni[1],M. Ferretti[1],S. Levialdi[2],R. Stefanelli[3]

1-Department of Computer and System Science - Universita' di Pavia
Strada Nuova 106/c - 27100 Pavia - Italy
2-Department of Mathematics - Universita' di ROMA
P.zza A. Moro 2 - 00185 Roma - Italy
3-Department of Electronics - Politecnico di Milano
P.zza L. da Vinci 32, I-20133 Milano - Italy

ABSTRACT

In 1981 a national research program for the design, simulation and construction of a multiprocessor image processing system was started. After a first phase devoted to the comparison of suggested and existing systems and to the definition of a set of benchmarks and to the evaluation of the performances of the major classes of machines, a new system has been defined. The structure of the new system is based on a pyramid of processors and many applications in which this machine may be exploited are highlighted. The multiprocessor architecture has been fully designed and the chip will be built by an Italian silicon foundry, the SGS company, within the framework of the multichip national project.

INTRODUCTION

In the last decade a high number of image processing systems have been proposed, designed and sometimes built. Several reviews of such systems can be found in the literature (1.6). In 1981 research groups belonging to a few italian universities proposed the design, the simulation and the construction of a new system for image processing to be inserted in the research project of national interest of the italian Ministry of Education.

Among the existing and proposed systems for "low level" image processing, two main classes of machines have been suggested: the array of processors (belonging to the SIMD class: Single Instruction stream / Multiple Data streams) see for example (7), and the pipeline of processors (for this particular application following properly the MISD class: Multiple Instruction streams / Single Data stream) see for instance the Cytocomputer (8). A performance comparison between these two classes (9) showed that for image processing tasks the first solution is more suitable even if the amount of time devoted to the "communication" is even higher than the proper computation time. In order to overcome this problem some proposals were presented in the literature (10); we selected the solution based on the pyramid structure.

A pyramid structure, see fig. 1, may be considered as the three dimensional extension of the two dimensional binary tree. Three main features of this structure suggest its usefulness in image processing.

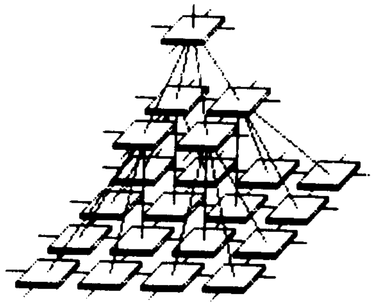


Fig. 1 Three layered pyramid structure

The first one relates to the topology of the pyramid.For the graph corresponding to this structure, having a given degree (maximum number of connections of a node in the graph) the diameter is nearly minimal, i.e. the path required to connect any pair of non adjacent nodes crosses only few intermediate nodes (11). Due to these interconnection characteristics the message passing capabilities change from O(N) (for N linear size of the array) to O(log N).

The second feature stems from the possibility of processing the same image at different resolution levels. Some years ago, a general strategy for image feature extraction was suggested, where the use of a low resolution version of the image produces a computation speed up (12). The planning approach formalizes this concept into an algorithmic framework (13).

The third and final feature is connected to the process of image analysis and description whereby the amount of data involved in the computation is gradually reduced in a cone like fashion (14). Looking at the pyramid along its axis and considering each succeeding level, each array contains the information extracted from the previous one. According to this view, the pyramid structure may be seen as a pipeline of array processors so allowing concurrent execution of instructions.

CH2146-9/85/0000/0237$01.00 © 1985 IEEE

The total hardware that is required, when compared to the one needed for the base array, can be evaluated by noting that each new plane increments the number of processors by 1/4 recursively until the apex of the pyramid. The total sum of the extra processors amounts to 1/3 of those present in the base.

By exploiting the three advantages stated above in some tasks a speed up factor of (N1/21ogN) may be obtained as can be seen in (15) where N is the linear size of the image and 1 is the number of bits per pixel. (This speed up is referred to the flat array).

In what follows a discussion of the main features of the new system are presented: the image distribution on the processors, the processor interconnection scheme, the bit serial arithmetic, the global features of the system and the input output capability.

The present version was designed on the basis of some technological limitations as for instance the maximum number of pins but a more mature version of the system is already under way.

PRELIMINARY CONSIDERATIONS

In practical applications a number of different image sizes are used: $(10^2)^2$ in the robotic field, $(10^3...10^4)^2$ in satellite imagery, $(512)^2$ in automatic inspection based on TV camera technology; etc. The amount of processing elements (PEs) considered up to now is 128 x 128 in the MPP (Massively Parallel Processor) (16) developed by Goodyear Aerospace for NASA. Referring to SIMD machines in practice the array of PEs is usually smaller than the image size.

In order to overcome this problem two solutions have been developed: to distributed the PEs over the image or to distribute the image over the PEs (17, 18). These two solutions lead towards very different processor structures and capabilities: in the former case the single processor deals with a block of the image, in the latter case each processor deals with just one pixel at a time.

In the latter case the processor is simpler and therefore a number of PEs can be contained in a chip, so exploiting the VLSI silicon technology, higher PE array sizes can be obtained; in this connection single bit processors are considered and only small amounts of local memory are supported.

Adopting the former approach the advantages of subnanosecond technologies (as gallium arsenide, Josephson effect) can be extensively exploited.

We chose the second approach and using a 4 µm Si gate NMOS technology five PEs have been included in the chip with 256 x 5 bits of RAM memory. The chip has been packed in a 48 pin dual in line module and the integration level of such chip is of the order of 30.000 transistors in an area of 5*5 mm. The five PEs constitute the elementary pyramid, one of them being the father of the other four sons forming the base. The features of the processors are non conventional when compared to the processor in the chips designed for SIMD machines which generally contained a higher number of PEs. The processors in our chip can be considered more powerful and some typical tasks for image processing, as local operations, can be naturally implemented.

A pyramid machine, hosted by a sequential computer, requires a new, sophisticated, programming environment to give the user the full availability of all the processing planes of the pyramid. A radically different operating system and high level language must be developed in order to obtain the required facilities. We are gradually progressing along this direction so as to obtain a fully usable image processing system. Many efforts are devoted to the development of high level languages for robotics, image analysis and more generally, for automation as documented in (19).

GLOBAL FEATURES

The pyramid structure because of the way in which an elementary pyramid is built (father and four sons) must be controlled so that a common instruction is executed on each couple of layers. In this way a Multi SIMD architecture is implemented and this system may generalize to a single instruction per layer providing two separate inputs on the chip driven by different controllers.

If all the PEs execute the same instruction the pyramid follows the natural SIMD mode but due to a layer masking facility planes may be inhibited and, furthermore, a PE masking register may disconnect a chosen subset of all the active PEs. In this way a flexible behaviour of the machine may be programmed so exhibiting a data dependent modality. Some image processing tasks use algorithms which take advantage of this feature and therefore overcome the limitations of the low level vision computations.

In parallel execution of image processing algorithms a control structure which may establish the termination of a recursive instruction(s) can be usually employed and is physically implemented by means of a global boolean OR having as inputs all the single state conditions of the PEs belonging to a single layer. This facility is called OR sum tree and may be graphically sketched as in figure 2.

As an example the output of an EX OR function between the present and previous states of all PEs allows the detection of a stable state in the layer (or in the unmasked PEs of the layer).

THE INPUT OUTPUT SYSTEM

The speed up factor of array processors over sequential machines is largely due to the parallel tightly orchestrated processing of large quantities of data held in the PEs local memories. Such an advantage would be reduced if the I/O process slowed down the overall computation.

The I/O subsystem must be able to input images while processing goes on and should adapt to different I/O rates and image sizes imposed from external devices. The solution we adopted is well known and allows for complete overlapping of I/O and processing and for bit column parallel loading

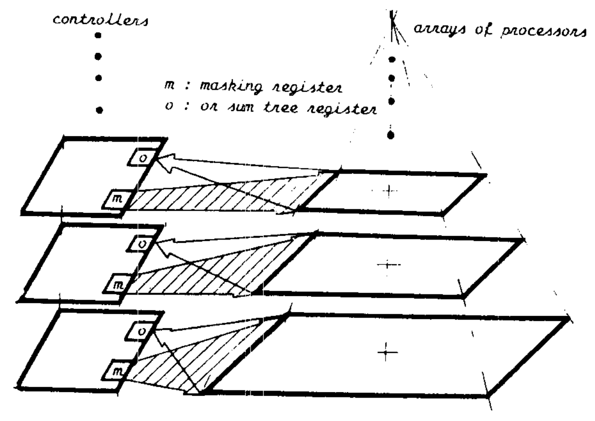


Fig. 2 Schematic diagram of a pyramid structure showing the masking and or sum tree controller capabilities

and unloading of images on every plane of the pyramid. Within each chip the four PEs of lower plane are equipped with a single bit register devoted to I/O; these registers are connected in columnwise ordered and are synchronized by a second clock in the system.

Images enter the pyramid at the desired plane in column parallel fashion; meanwhile processing goes on as dictated by the first main clock and is interrupted only for one cycle of the second clock to store a complete bit plane in the PEs local memories. The pyramid can load a new image, process previously loaded images at the same and/or different layers and provide results at the final plane: all these activities being completely overlapped.

The second clock of the system, which is responsible of these facilities can also be driven at different rates to match the speed of the external devices linked to the pyramid.

It is generally useful to have an interface buffer memory between the acquisition system and the pyramid system so as: to provide the image data in a bit plane format and to match the image/subimage size to the layer sizes.

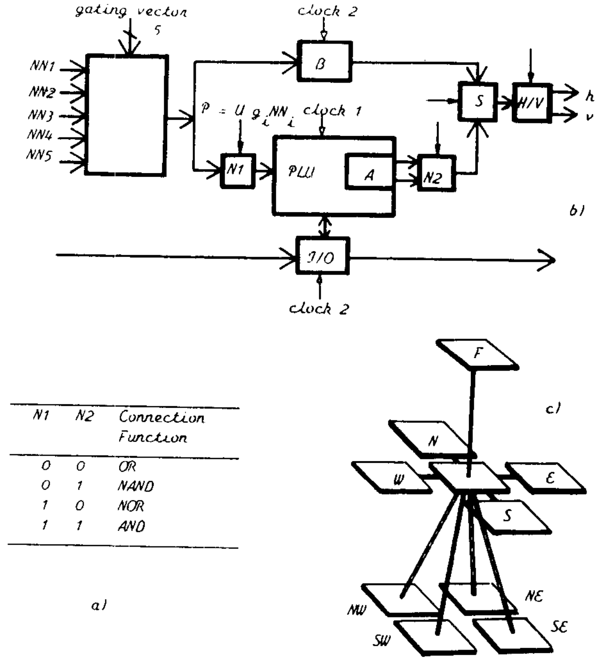
THE INTERCONNECTION SCHEME

Several tessellation modes have been proposed in the SIMD family: the four connected one (MPP, DAP, LIPP, etc.); the six connected one (GLOPR, diff3, etc.); the eight connected one (CLIP4, PHP II,Base 8, etc.). Besides the choice of the connectivity another choice can be made regarding the neighbor information access: gating or multiplexing. The first alternative allows a simultaneous access to a subset of the neighbors (implementing a particular boolean function), in the second one only one neighbor at a time may be accessed.

In our case we have to consider two kinds of interconnections: a "horizontal" one relates to the communication among PEs of the same layer, a "vertical" one links PEs of different layers: for this system four sons in the preceeding layer and one father in the successive layer.

The implementation of common algorithms on a simulator of our structure showed that the need of a selection of a subset of the horizontal (H) neighbors together to a subset of vertical (V) neighbors was quite unusual, then we implemented a gating technique in which pins are shared so that H and V selections are disjoined, i.e. two machine cycles are needed if a union of H and V subsets is required.

Moreover due to the constraints on the number of pins in this implementation (48 pins dual in line) only the four horizontal connectivity has been realized. Four possible logical functions have been implemented among the selected neighbors: namely the OR, AND, NOR, and NAND functions (see figure 3a).



N1	N2	Connection Function
0	0	OR
0	1	NAND
1	0	NOR
1	1	AND

Fig. 3 a: neighborhood connection functions; b: simplified block scheme of PE core; c: vertical and horizontal neighborhood of a PE

It is well known that for low level image processing the local operations are largely the most common ones (it has been said that the neighborhood access problem is the Von Neumann bottleneck of image processing (18)). Moreover the subarray of interest cannot be limited to the three by three window. Iterative three by three local operations can produce larger windows of the local region of interest, but introducing some constraints in the operations (20).

For this reason and also to increase the speed of the other common operations (as for instance the one of extracting from the apex the content of a PE of the base and viceversa) a new communication facility has been introduced. One of the two main registers of the PE can be used, exploiting the gating system for the neighborhood(s) selection, as a shift register distributed in the pyramid. In this way, using an auxiliary clock, a fast long communication between non adjacent PEs can be performed (a similar solution is used in CLIP4 and DAP (21)), refer to figure 3b.

The input output circuitry may also be used to enable one way horizontal interprocessor communication concurrently with computation as described above.

THE SERIAL ARITHMETIC

The major reason which lead to serial arithmetic in all SIMD multiprocessor systems for image processing is the need to keep the PE architecture as simple as possible. This allows the integration of more PEs in a single chip thus reducing the interconnection complexity at board level and so covering the maximum part of the image. Most tasks require binary image processing, neighborhood connections can be realized practically with one bit paths so economizing silicon area. We have chosen serial arithmetic for the above reasons, yet the processing of L grey levels images can be easily accomplished working serially on the $1=\log_2L$ bit planes of the image.

A wide variety of PEs architectures is available: all share boolean processing capability while only a small fraction of them are equally suited for efficiently performing arithmetic tasks.

Our design aims at the construction of a pyramid architecture both at chip and at board levels. In the actual implementation five PEs are included in a chip; a future version might lead to a 21 PEs per chip solution. We have exploited the possibility to augment the PE capability over that of preceeding SIMD machines (usually based on 8 PEs per chip).

We have chosen to introduce in each PE two variable length shift registers, SR1 (the highest one in fig. 4) and SR2 (the lower one in figure 4). Both registers can be used as inputs to the ALU; SR2 works as an accumulator, SR1 as a link in the gating logic and can be used to supply the neighboring values involved in the computation.

The variable length of the shift registers is used to match the computations to the number of bits per pixel of the image.

The ALU of the PE is composed of:

- a two input boolean unit which works on the two single bit registers A and B and provides the result in A;
- a comparator on the contents of the two shift registers SR1 and SR2 which gives the result of one of the three possible tests in A;
- a full-adder for integer addition of the content of SR1 and SR2 providing the result in SR2

Local operations are speeded up by this architecture: partial results need not be stored in the local RAM, since they are kept in SR2. Logical operations and integer addition can be accomplished by the existing hardware.

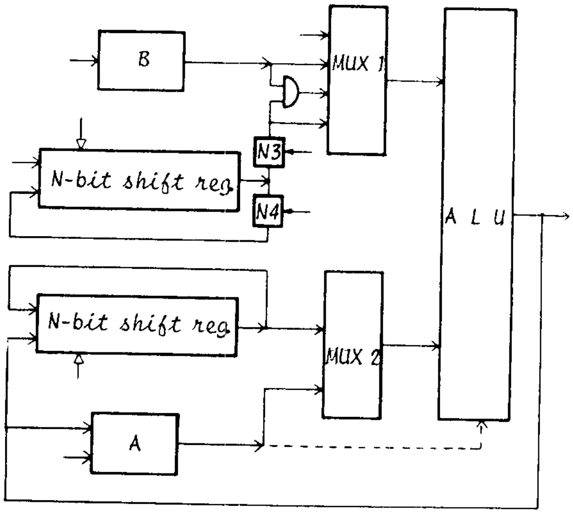


Fig. 4 Simplified block scheme of the PE core showing its computational capabilities

As an example the addition of n bit numbers requires the loading of the two numbers in the shift registers (SR1 and SR2) and n shift cycles to obtain the final result in SR2.

Multiplication instead, is performed by introducing the multiplicand in SR1 and, bit by bit, the multiplier in B. For each bit a logical product of the contents of SR1 and B is computed by hardware and added to the content of the accumulator SR2; a final logical right shift is performed to properly allign the partial results.

This procedure requires n^2 cycles whilst if the multiplier is a constant for the whole array the time dependence falls, on the average, by 1/2.

EXAMPLE OF TASK EXECUTION BY PAPIA

In order to demonstrate the capabilities of PAPIA's architecture on image processing tasks we will describe the corresponding program schemata for two typical IP tasks, namely edge extraction by a local operator and histogramming.

Although the speed of the processing unit of a new computing architecture is generally described in terms of MIPS or MFLOPS, the PAPIA machine oriented to image processing tasks is better described by means of the execution of typical operations on images, i.e. local operations and statistics (9).

As well known, local operations are the basis of many image preprocessing and processing tasks and generally consist of the computation of a new pixel value as a function of the values of neighboring pixels.

EDGE EXTRACTOR

A very popular edge extractor is the Sobel operator based on a three by three subarray having a set of five valued weights as shown in fig. 5

$$\begin{matrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{matrix} \qquad \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{matrix}$$

Fig. 5 Vertical and horizontal templates for the Sobel operator

The Sobel filter evaluates the new pixel value by convolving the input image by the two templates shown in fig. 5.

The following steps describe the PAPIA implementation by the Sobel operator.

1. Every PE will add (in accumulator SR2) the value of its north (west) and south (east) neighbor and the local pixel value multiplied by two (shifted one bit to the left). The pixel contents of SR2 are stored in the local memory.
2. The result of the previous step is shifted to the east (south) PE neighbor and stored in SR2.
3. The result of the first step is shifted to the west (north) PE neighbor and stored in SR1.
4. The subtraction between the contents of SR2 and SR1 is performed obtaining the result in SR2.
5. The contents of SR2 are compared with 0 and, if they are negative, (the contents of A is 1) they will be complemented by a SIMD operation on the array (by tranferring the A content to the PE masking register, only PEs with SR2 containing a negative value change sign).

Steps 1-5 must be repeated for the vertical operator substituting the directions indicated in brackets.

Taking care of all the described information transfers, the full task may be executed (for an image of up to 128*128 pixels) in about 38*1 clock cycles.

HISTOGRAM COMPUTATION

The second task is generally used in image analysis and requires to compute all the pixels having the same grey value

If we assume that the image is made of pixels having different grey values, and that the number of pixels on a row is equal to the number of grey values we may generate the histogram by performing the following steps:

1. A stepwise ramp having integer values (from 0 to 1) is simultaneously generated along each column and stored in SR2.
2. The pixel information contained in each column of the image (in SR1) is circulated in wrap around mode and at each step of this process a comparison is made between the contents of SR1 and SR2. The result is loaded in the A register.
3. Every value obtained at each comparison cycle is stored in a one bit stack in the PE local memory. This step and the previous one are repeated L times so obtaining the histogram for each column.
4. Within the SR2 register, each processor will add the contents of the one bit stack loading the mask register on a bit by bit basis so enabling the increment of the register.
5. In order to cumulate the column histogram along each row recursive doubling will be performed. The first L/2 columns are east shifted into the SR1 registers so covering the remaining L/2 columns. This process is repeated recursively with partitions L/4, L/8, etc. so obtaining at the end the full histogram on the last column.

The full task may be executed by a PAPIA architecture in about 7*N*1 clock cycles (about 7.000 clock cycles for a 128*128 image, 8 bits per pixel).

CONCLUSIONS

A general approach in the design of a multiprocessor architecture is to study the computational requirements of a set of application areas and then, in a top down strategy, the algorithms and operations used, finally an architecture that well matches such requirements (23).

Within the area of image processing, a performance analysis has been accomplished to verify and refine the architecture, essentially by simulation. As a consequence of this analysis, the following choices have been made: distributed image topology (one pixel per processor), multiresolution layered architecture, orthogonalization of vertical and horizontal operations, gating logic allowing neighborhood boolean operations and, lastly, variable length shift registers for speeding up arithmetical operations and typical tasks as convolution (24).

In the technological implementation using VLSI circuitry, three problems have been considered: the external data communications, the internal interprocessor communications and the faults management. In the first problem the trend has been to increase the ratio between the number of internal gates to the number of input output pins so having large computation power for each input data. The second problem is tackled by using a regular, modular and recursive structure so having a high ratio of computing chip area to internal communication area. Finally to solve the third problem extra chip reconfiguration and addition of components have been the two main strategies to correct a faulty result (22).

For many years researchers in image processing have tried to design software tools which make better use of the capabilities of computers. In our case, the programming of the pyramid structure in assembler language is unthinkable due to the complexity of the multi SIMD systems. For this reason research has to be directed towards the definition of a high level language (perhaps an extension of an existing one) that is able to manipulate naturally the objects of image processing, that may provide adequate control in the parallel execution of instructions and, finally, that may provide friendly interaction with the user when partial or final results must be inspected, see for instance (23).

ACKNOWLEDGEMENTS

The authors wish to thank Prof. F. Maloberti for his help in the electronic layout and A. Canobbio, L. Cinque, G. Varasano for their continuous cooperation towards the realization of this project.

REFERENCES

[1] P.E. Danielsson, S. Levialdi, "Computer Architectures for Pictorial Information Systems", IEEE Computer, vol. 14, n. 11, 1981, pp. 53-68.

[2] K. Preston, Jr., "Cellular Logic Computers for Pattern Recognition", IEEE Computer, vol. 15, n. 1, 1983, pp. 36-47.

[3] M. Kidode, "Image Processing Machines in Japan", IEEE Computer, vol. 15, n. 1, 1983, pp. 68-80.

[4] V. Cantoni, S. Levialdi, "Matching the task to an image processing architecture", Computer Vision, Graphics and Image Processing, 22, 1983, pp. 301-309.

[5] M.J.B. Duff, "Special hardware for pattern processing", 6th IJCPR, Munich, 1982, pp. 301-309.

[6] V. Cantoni, "Classification schemes for image processing architectures", NATO ASI on Computer Architectures for Spatially Distributed Data, Cetraro, 1983.

[7] M.J.B. Duff, "Review of the CLIP image processing system", AFIPS Conf. Proc. vol. 47, 1978, NCC, pp. 1055-1060.

[8] R.M. Lougheed, D.L. McCubbrey, S.R. Sternberg, "Cytocomputer: architectures for parallel image processing", Proc. Workshop Picture Data Descr. and Management, Pacific Grove, California, 1980, pp. 281-286.

[9] V. Cantoni, C. Guerra, S. Levialdi, "Towards an evaluation of an image processing system", M.J.B. Duff, Ed., Computing Structures for Image Processing, Academic Press, London, 1983, pp. 43-56.

[10] M.Onoe, K. Preston, Jr., A. Rosenfeld, Eds., Real Time / Parallel Computing, Plenum Press, New York, 1981.

[11] L. Uhr, "Pyramid Multicomputer Structures, and Augmented Pyramids", in M.J.B. Duff, Ed., Computing Structures for Image Processing, Academic Press, London, 1983, pp. 95-112.

[12] S. Tanimoto and A. Klinger, Eds., Structured Computer Vision: Machine Perception Through Hierarchical Computation Structures, Academic Press, New York, 1980.

[13] M.D. Kelly, "Edge detection in pictures by computer using planning", in Machine Intelligence 6, Edinburgh, University Press, 1971, pp. 397-409.

[14] L. Uhr, "Layered 'recognition cone' networks that preprocess, classify and describe", IEEE Trans. Comp. C 21, 1972, pp. 758-768.

[15] V. Cantoni, L. Carrioli, L. Cinque, M. Ferretti, S. Levialdi, A. Perone, "Image processing in hierarchical computer structures", International Conf. on Digital Signal Processing, Florence (Italy), 1984.

[16] K.E. Batcher, "Design of a Massively Parallel Processor", IEEE Trans. Comp. C 29, 1980, pp. 836-840.

[17] A. Rosenfeld, "Parallel Image Processing using Cellular Arrays", IEEE Computer, vol. 16, n. 1, 1983, pp. 14-20.

[18] P.E. Danielsson, "Vices and virtues of image parallel machines" in Digital Image Analysis, S. Levialdi, Ed., Pitman, London, 1984, pp. 47-59.

[19] Proceedings of IEEE Workshop on "Languages for Automation", Chicago, 1983, IEEE Comp. Soc. n. 506, pp. 1-311.

[20] B. Kruse, "PICAP and relational neighborhood processing in FIP", in Multicomputers and Image Processing, K. Preston, Jr., L. Uhr, Eds. Academic Press, New York, 1982, pp. 31-46.

[21] A.P. Reeves, "On efficient global feature extraction methods for parallel processing", Computer Graphics and Image Processing, vol. 14, 1980, pp. 159-169.

[22] M.G. Sami, R. Stefanelli, "Reconfigurable Architecture for VLSI Processing Arrays", National Comp. Conf., AFIPS, Anneheim, 1983.

[23] S. Levialdi, "Computer Architectures for Image Processing", panel position, IFIPS, Paris, 1983.

[24] V. Cantoni, M. Ferretti, S. Levialdi, F. Maloberti, "A pyramid project using integrated technology", in Integrated Technology for Parallel Image Processing, Ed., S. Levialdi, Academic Press, New York, (in press).