PAPRICA-3
A REAL-TIME MORPHOLOGICAL IMAGE PROCESSOR

A.Broggi, G.Conte

Dip. di Ingegneria dell'Informazione
Università di Parma, Italy

G.Burzio

CRF, Centro Ricerche FIAT
Torino, Italy

L.Lavagno, F.Gregoretti, C.Sansoè

Dip. di Elettronica
Politecnico di Torino, Italy

L.M.Reyneri

Dip. di Ingegneria dell'Informazione
Università di Pisa, Italy

ABSTRACT

The paper describes the architecture of an image coprocessor based on a linear array of elementary processors. The system has been designed for real-time vision applications and embeds a direct I/O interface to a camera device.

The instruction set of the elementary processors is based on morphological match operators on a reduced 5×5 neighborhood.

Two additional communication mechanisms, one fixed and one programmable, provide both global and multiple communication channels among processing elements which are not directly connected.

1. INTRODUCTION

PAPRICA-3 is a SIMD massively parallel system devoted to Image Processing tasks with an instruction set oriented to mathematical morphology [8, 9]. It has been designed as an embedded system for applications with hard real time requirements such as obstacle avoidance and lane keeping in the automotive field [2, 1] or visual inspection and object recognition in robotics. The key issues of these applications are the fast response time of the system and the high computational power required together with the reduced overall system cost, power consumption, and space required.

This work was partially supported by CNR Progetto Finalizzato Trasporti under contracts 93.01813.PF74, 93.04759.ST74 and 91.01034.PF93.

The system has been designed on the basis of the experience gained in the implementation of a first prototype architecture, PAPRICA [6, 4] with a bidimensional grid interconnection and operating as a coprocessor of a general purpose workstation. The performance issues which have been extensively analyzed in [5] have led to the choice of a linear array architecture [7] and to the decision to integrate the I/O interface directly in the system as shown in the block diagram of figure 1. A camera is directly interfaced to the Processing Array (PA) whose output may be directly connected to a display device or to a Host Processor for further processing. The Host is also responsible for transferring programs to the array.

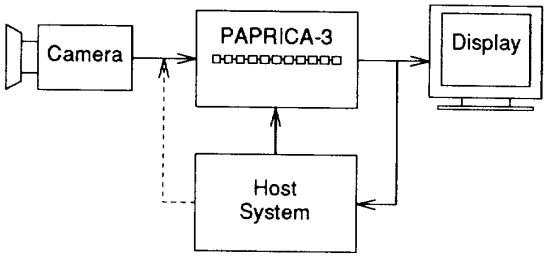

Figure 1: Block diagram of the system

Two versions of the system are foreseen, one for field use and one for application development. In the former the Host will be a fast DSP system which will handle iconic to symbolic processing and high level management tasks. In the latter the Host will be a general purpose workstation and an additional communication

0-8186-6950-0/94 $4.00

channel (dashed in figure 1) will be available to load the PA from the Host Memory for debugging purposes.

2. SYSTEM DESCRIPTION

As shown in figure 2a, the core of the processor is a dedicated SIMD cellular architecture based on a linear array of Q identical 1-bit Processing Elements (PEs). The array as a whole is connected to an external image memory via a bidirectional Q-bit bus, therefore each memory read or write operation transfers a complete vector of Q pixels, 1 bit per pixel, at a time. The *Image Memory (IM)* contains one or more *Images*, each one being made of H lines of length Q pixels. The rationale behind this system is that the size of the PA matches exactly the width of the input image, i.e. the width of the imager device. This solution reduces the PE virtualization mechanism problem, which has been proven to be a critical design issue [5]. Therefore the value of Q will range from a minimum of 128 to 1024 if the implementation constraints, namely the cost, will allow. The PA processes one full image line per machine cycle, whose duration is T_C (about 10 ns). Since the PEs implement a *bit-serial approach*, at each cycle the PA executes morphological and boolean operations just on 2 or 3 layers at a time, while for more complex computations several machine cycles are needed both for operand loading and for processing (e.g. 8 bit add or subtract requires 8 cycles, while 8 bit multiplications need up to 64 cycles).

Data are transferred into the PEs internal registers, processed, and explicitly stored back into memory according to a RISC-oriented processing paradigm. Since the instruction set is based on morphological operators, the result of an operation depends, for each processor, on the values of pixels in a given neighborhood (5×5). Data from **EAST** and **WEST** directions may be obtained by direct connection with neighboring PEs while all other directions correspond to data of previous (**N**,**NE**,**NW**) or future (**S**,**SE**,**SW**) data lines. For this reason a number of processor registers (*Morphological Registers (MOR)*) have a structure which is more complex than that of a simple memory cell and are in reality composed of 5 1-bit cells with a **S→N** shift register connection. Besides the MOR, each PE owns also a number of *Logical Registers (LOR)* which may be used for intermediate storage and for all operations which do not require the use of neighboring values.

The system comprehends also a Serial-to-Parallel I/O Device, called *Imager Interface*, which can be connected to a conventional camera. While a line is processed, the Imager Interface automatically loads the following image line from the camera. At the end of the processing, the PA stores the results back into the Imager Interface (on different bit-planes) and loads in parallel the following image line. During the data acquisition process, the Imager Interface behaves like a shift-register, loading the data serially from the camera, and outputs the processed data serially to a monitor.

It is well known [3] that, due to the restricted neighborhood accessed by the PEs, a program needs to be split into different blocks, and each block must be applied to the whole image, one line at a time. An algorithm for the optimal partitioning of an Assembly program is presented in [3]. In order to speedup the program fetching from the Program Memory, the system includes also a *Writable Control Store (WCS)* (internal to the PA) that is used to store each block of instructions, for a faster access.

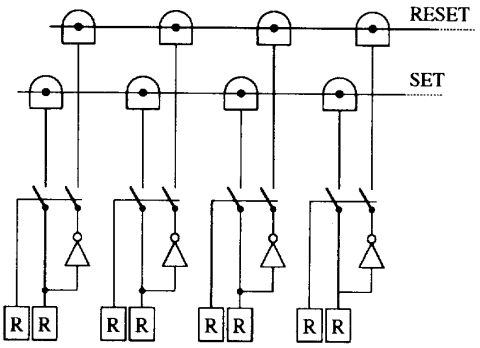

Figure 3: Flag Evaluation Network (FEN)

Two logical interprocessor communication mechanisms are available for exchanging information among PEs which are not directly connected. The first one (the Flag Evaluation Network, FEN), shown in figure 3, is global the whole PA and provides two **SET**/**RESET** Flags, reflecting the status of the central pixel all PEs: all '1' or all '0', respectively. These two flags are used in control flow constructs handled by the PEs.

The second one (the Interprocessor Communication Network, ICN), shown in figure 4, allows global and multiple communications among components of different subsets of the PA (clusters of PEs) and its interconnection topology is programmable. In fact, each PE drives a switch that enables or disables the communication between them; the PEs can thus be dynamically grouped into clusters in which each PE can broadcast its values to the whole cluster within a single instruction. This feature can be extremely useful in algorithms

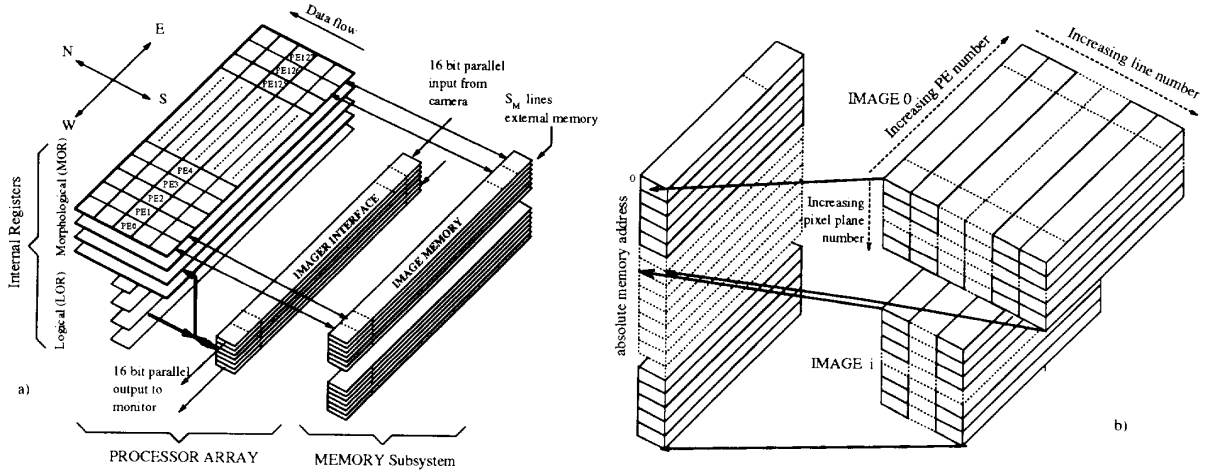

Figure 2: Structure of the Processor Array and of the Memory allocation

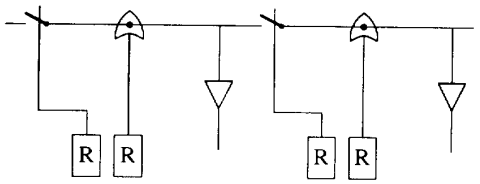

Figure 4: Interprocessor Communication Network (ICN)

involving *seed-propagation* techniques, and in the emulation of pyramidal (hierarchical) processings.

3. IMAGE MEMORY ORGANIZATION

As shown in figure 5, the PA is tightly interfaced to the Image Memory via a bidirectional bus. Bus parallelism must match the number Q of PEs, in order to exploit the full bandwidth of the communication channel. For smaller (e.g. embedded) systems, a multiplexed bus can also be used but this impairs the overall performance. Due to the large parallelism (at present $Q = 128$ to 1024), a number of 32-bit memory modules is used. All modules share the same address lines and they therefore span the same address space (currently 64K to 1M words of Q bits each).

From the physical point of view, the IM can be seen as individually addressable lines of Q bits (*absolute line address*). However, this raw memory is organized into more complex logical structures, as shown in figure 2.b, which may vary at run-time. As an example, lines can be arranged into *bit-planes* (i.e. a collection of adjacent lines) or into gray-scale or color images (i.e. a collection of bit-planes). Mapping between logical data structures and physical addresses is performed by a *Controller* (CT) external to the PA (see figure 5).

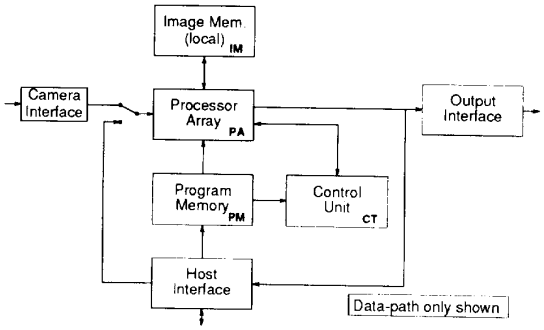

Figure 5: Internal architecture of the PAPRICA-3 system

The memory speed is still a system bottleneck, since the internal clock period can be as small as 10 ns, while the cycle time of unexpensive memory modules is usually 3 to 5 times higher. This implies that memory transfer instructions shall insert a number of wait cycles.

According to the applications foreseen for the system, a different memory space can be envisaged. A small, low-cost embedded system can have as few as 32K×64 bits, where a total of sixtyfour 8-bit images, 64 × 64 pixels of size can be stored. Larger systems

can have as much as 8M×256 bits, where up to forty 24-bit full color images, 2048 × 1024 pixels of size can be stored.

A large memory space is envisaged for high-definition images (e.g. for real-time HDTV applications). This provides much higher performance at the expense of a higher cost.

4. INSTRUCTION SET

Beside *Initialization*, *Memory Transfer*, and *Flow Control* Instructions, PAPRICA-3 features a *Morphological* Instruction Set based on a reduced 5 × 5 neighborhood [9]. Morphological Operators are parallel instructions which operate on a given "neighborhood" of each PE, and are based on *Matching Operators* derived from the *hit-or-miss* transform proposed by [9] and on its extension to pyramidal architectures.

A *Matching Template* is a 5×5 pixels ternary pattern ('1', '0', or 'don't care'), against which the neighborhood of each PE is compared for matching. The result of the operator is a boolean value which can either be true or false if the neighborhood matches or not the template, respectively.

An elementary PAPRICA instruction is the concatenation of a unary match operator `MOP` applied to a given bit-plane, with a binary *Logical Operator* `LOP`. The latter is applied to the `MOP`'s output and to another bit-plane of the central pixel.

The overall result may be either stored or accumulated (either OR-ed or AND-ed) into a destination register. The complete syntax of a generic morphological instruction is either of:

```
Rd   =  [MOP](Rs1) [LOP (Rs2)] [%EN]
Rd  |=  [MOP](Rs1) [LOP (Rs2)] [%EN]
Rd  &=  [MOP](Rs1) [LOP (Rs2)] [%EN]
```

where `Rd`, `Rs1`, and `Rs2` are three PE registers, namely the destination, the `MOP`'s and the `LOP`'s second input, respectively. `MOP` is a binary coding of the matching template, while `LOP` indicates one of eight boolean functions (e.g. NOT, AND, OR, etc.). The three versions of the syntax refer to direct storage, OR-accumulation and AND-accumulation, respectively.

The optional flag `%EN` acts as a Write Enable Mask, in the sense that, when present, the writing (or accumulating) process is conditioned to the corresponding bit of the mask.

The instruction set gives also provision for handling (i.e. computing and using) the three global flags and the interprocessor communication network. The flow control instructions can behave differently, according to the flag status.

5. SYSTEM IMPLEMENTATION

A first prototype of the system is currently being designed, to be implemented on a set of two ASIC chips. The first one (a full-custom design) will incorporate an array of either 64 or 128 PEs. The second one (a standard-cell, semi-custom design) will incorporate all the circuits for the controller CT.

Two or more PA chips can be cascaded to build an array of as many PEs as required by the application. Currently a value of Q between 64 and 1024 is envisaged. The PA chips, together with a number of commercial SIMM memory modules (currently $\frac{Q}{32}$ chips with 32 bits parallelism), the controller, imager and video interfaces and the host interface (or the host itself in an embedded system) can be incorporated together on a single PCB board (size will depend on the desired system size).

The controller chip handles all transfers to/from the memory, executes program flow control instructions, handles the synchronization and data transfer to/from a host computer (DSP) and synchronizes the imager and the video.

The current target technology is a double metal 1 μm CMOS. With that technology an internal clock rate or 100 MHz (i.e. cycle time T_C = 10 ns) is expected, by properly designing the PA chip. Pipelining and systolic techniques will be extensively used. A memory transfer cycle time T_M of 50 ns (i.e. a transfer rate of up to 5 Gbit/s for Q = 256 can also be obtained).

A reduced performance prototype, with no internal pipelining and lower speed memories, is also envisaged. This system will have an internal clock rate of 20 MHz and a memory cycle time T_M of 150 ns.

In the future more performant technologies (e.g. either 0.7 μm or 0.5 μm, triple metal) will provide an internal clock rate of up to 200 MHz and a number of PEs per chip up to 1024.

A Multi Chip Module (MCM) implementation is foreseen for further implementations, when very high performance is required in a very small space.

6. PERFORMANCES

Thanks to the proposed architecture, PAPRICA-3 fully exploits two main factors: the massive parallelism of the PA and the large width of the memory bus. Furthermore both factors are matched to the size of the im-

ALGORITHM	Number of bit/pixel	Small system (64 PE, 50/150 ns)	Large system (256 PE, 10/50 ns)
Contouring of B/W image	1	≈ 0.8 ns/pel	≈ 0.04 ns/pel
Skeleton of B/W image	1	$\approx 200 - 400$ ns/pel	$\approx 10 - 20$ ns/pel
Pattern Matching (5×5)	1	$\approx 0.8 - 1.6$ ns/pel	≈ 0.04 ns/pel
Matrix Summation	8	≈ 7 ns/pel	≈ 0.4 ns/pel
3×3 pixel Average	8	≈ 30 ns/pel	≈ 1.6 ns/pel
5×5 pixel Average	8	≈ 60 ns/pel	≈ 3.2 ns/pel
Optical Flow (max. ± 8 pels per direction, full search)	8	≈ 5.0 μs/pel	≈ 80 ns/pel

Table 1: Estimated performance figures of PAPRICA-3 system (values between brackets are Q, T_C/T_M)

ager, since from preliminary analyses this seems a good compromise among system complexity, speed, chip pinout limitations, and flexibility. The feasibility of the system and a forecast of its performances have been obtained through a software system simulator, currently running on Unix Workstations, but as in the case of the first prototype [4, 5], a more efficient version of the simulator will be developed on a Connection Machine CM-2.

The expected performance of the proposed architecture depend mainly on the specific application. The tasks exploiting the maximum performance are "cellular" or "morphological" algorithms, which require limited communications among PEs which are far apart from each other. A few performance figures for simple examples are presented in table 1, assuming a worst case and a best case system composed respectively of 64 and 256 processing elements, with a processing cycle time of 50 ns (10 ns) and a memory cycle time of 150 ns (50 ns).

7. REFERENCES

[1] G. Adorni, A. Broggi, G. Conte, and V. D'Andrea. A self-tuning system for real-time Optical Flow detection. In *Proceedings IEEE System, Man, and Cybernetics Conference*, volume 3, pages 7–12, Le Toquet, France, October 17-20 1993.

[2] A. Broggi. Parallel and Local Feature Extraction: a Real-Time Approach to Road Boundary Detection. *IEEE Transactions on Image Processing*, February 1995. In press.

[3] A. Broggi. Performance Optimization on Low-Cost Cellular Array Processors. In *Proceedings MPCS - IEEE International Conference on Massively Parallel Computing Systems*, Ischia, Italy, May 2-4 1994. EuroMicro - IEEE Computer Society. In press.

[4] A. Broggi, G. Conte, F. Gregoretti, C. Sansoè, and L. M. Reyneri. The PAPRICA Massively Parallel Processor. In *Proceedings MPCS - IEEE International Conference on Massively Parallel Computing Systems*, Ischia, Italy, May 2-4 1994. EuroMicro - IEEE Computer Society. In press.

[5] F. Gregoretti, L. M. Reyneri, C. Sansoè, A. Broggi, and G. Conte. The PAPRICA SIMD array: critical reviews and perspectives. In L. Dadda and B. Wah, editors, *Proceedings ASAP'93 - IEEE Computer Society International Conference on Application Specific Array Processors*, pages 309–320, Venezia, Italy, October 25-27 1993.

[6] F. Gregoretti, L. M. Reyneri, C. Sansoè, and L. Rigazio. A chip set implementation of a parallel cellular architecture. *Microprocessing and Microprogramming*, 35:417–425, 1992.

[7] L. A. Schmitt and S. S. Wilson. The AIS-5000 Parallel Processor. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 10(3):320–330, May 1988.

[8] R. M. Haralick, S. R. Sternberg, and X. Zhuang. Image Analysis Using Mathematical Morphology. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 9(4):532–550, 1987.

[9] J. Serra. *Image Analysis and Mathematical Morphology*. Academic Press, London, 1982.