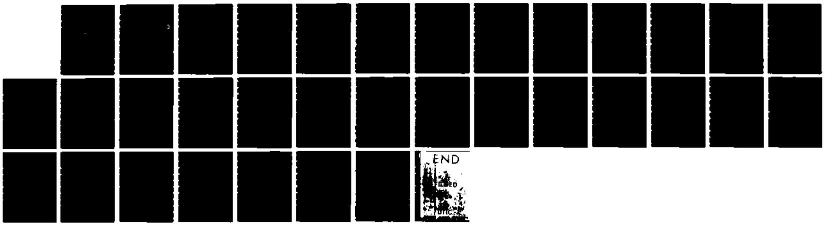
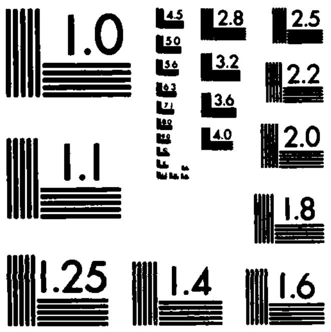
AD-A136 292   A VERSATILE PARALLEL IMAGE PROCESSOR SYSTEM(U) PURDUE   1/1
UNIV LAFAYETTE IN DEPT OF ELECTRICAL ENGINEERING
H J SIEGEL OCT 83 AFOSR-TR-83-1231 AFOSR-78-3581

UNCLASSIFIED   F/G 9/2   NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AFOSR-TR- 83-1231

(4)

A136292

A VERSATILE PARALLEL IMAGE PROCESSOR SYSTEM: FINAL REPORT

Howard Jay Siegel
School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

October 1983

DTIC
ELECTE
DEC 22 1983
D
A

This report describes the research supported by the Air Force Office of Scientific Research grant AFOSR-78-3581, during the period March 1, 1978 to December 31, 1982. The main thrust of the research was the design of "PASM."

PASM is a large-scale multimicroprocessor system being designed at Purdue University for image processing and pattern recognition. This system can be dynamically reconfigured to operate as one or more independent SIMD and/or MIMD machines. PASM consists of a parallel computation unit, which contains N processors, N memories, and an interconnection network; Q micro controllers, each of which controls N/Q processors; N/Q parallel secondary storage devices; a distributed memory management system; and a system control unit, to coordinate the other system components. Possible values for N and Q are 1024 and 16, respectively.

This report consists of two parts. The first is an overview of the PASM system. It is a preprint of a paper to appear as a chapter in *Computer Architectures for Spatially Distributed Data*, H. Freeman and G. G. Pieroni, editors, Springer-Verlag, New York, NY, 1983. In this paper the interconnection network, control schemes, and memory management in PASM are described. Examples of how PASM can be used to perform image processing tasks are also given. The second part of the final report is a list of the 53 publications that describe in detail the research that was supported by this grant.

DTIC FILE COPY

83 12 22 031

Approved for public release;
distribution unlimited.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR- 83-1231	2. GOVT ACCESSION NO. AD-A136	3. RECIPIENT'S CATALOG NUMBER 292
4. TITLE (and Subtitle) A VERSATILE PARALLEL IMAGE PROCESSOR SYSTEM: FINAL REPORT		5. TYPE OF REPORT & PERIOD COVERED FINAL, 1 MAR 78-31 DEC 82
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Howard Jay Siegel		8. CONTRACT OR GRANT NUMBER(s) AFOSR-78-3581
9. PERFORMING ORGANIZATION NAME AND ADDRESS School of Electrical Engineering Purdue University West Lafayette IN 47907		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS PE61102F; 2304/A2
11. CONTROLLING OFFICE NAME AND ADDRESS Mathematical & Information Sciences Directorate Air Force Office of Scientific Research /NM Bolling AFB DC 20332		12. REPORT DATE OCT 83
		13. NUMBER OF PAGES 31
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Image processing, interconnection networks, memory management, MIMD machines, multimicroprocessor systems, multiple-SIMD machines, parallel processing, partitionable computer systems, PASM, reconfigurable computer systems, SIMD machines.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

PASM, a large-scale multimicroprocessor system being designed at Purdue University for image processing and pattern recognition, is described. This system can be dynamically reconfigured to operate as one or more independent SIMD and/or MIMD machines. PASM consists of a parallel computation unit, which contains N processors, N memories, and an interconnection network; Q micro controllers, each of which controls N/Q processors; N/Q parallel secondary storage devices; a distributed memory management system; and a system control unit, to coordinate the other system components. Possible values for N and Q are 1024 (CONTINUED)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

83 12 22 031

ITEM #20, CONTINUED: and 16, respectively. The interconnection network, control schemes, and memory management in PASM are described. Examples of how PASM can be used to perform image processing tasks are given.

In addition, there is a list of 53 publications that describe in detail the research that has been supported by this grant.

THE PASM SYSTEM AND PARALLEL IMAGE PROCESSING

Howard Jay Siegel
School of Electrical Engineering
Purdue University
West Lafayette, IN 47907 USA

To appear in:

Computer Architectures for Spatially Distributed Data, H. Freeman and G. G. Pieroni, editors, Springer-Verlag, New York, NY, 1983.

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFSC)
NOTICE OF TRANSMITTAL TO DTIC
This technical report has been reviewed and is
approved for public release IAW AFR 190-12.
Distribution is unlimited.
MATTHEW J. KERPER
Chief, Technical Information Division

1. Introduction

One way to do image processing faster is through the use of parallelism. Different modes of parallelism can be employed in a computer system. The SIMD (single instruction stream - multiple data stream) mode [9] typically uses a set of N processors, N memories, an interconnection network, and a control unit (e.g., Illiac IV [6], STARAN [5], CLIP4 [8], MPP [16]). The control unit broadcasts instructions to the processors and all active ("enabled") processors execute the same instruction at the same time. Each processor executes instructions using data taken from a memory with which only it is associated. The interconnection network allows interprocessor communication. An MSIMD (multiple-SIMD) system is a parallel processing system which can be structured as one or more independent SIMD machines (e.g., MAP [13]). The Illiac IV was originally designed as an MSIMD system [3]. The MIMD (multiple instruction stream - multiple data stream) mode [9] typically consists of N processors and N memories, where each processor can follow an independent instruction stream (e.g., C.mmp [27], Cm* [25]). As with SIMD architectures, there is a multiple data stream and an interconnection network. A partitionable SIMD/MIMD system is a parallel processing system which can be structured as one or more independent SIMD and/or MIMD machines (e.g., TRAC [17]).

This research was supported by the Defense Mapping Agency, monitored by the United States Air Force Command, Rome Air Development Center, under contract number F30602-C-0193; and by the Air Force Office of Scientific Research, Air Force Systems Command, USAF, under grant number AFOSR-78-3581. The United States Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation hereon.

In this paper, the organization of PASM [20], a partitionable SIMD/MIMD system being designed at Purdue University, is overviewed. Example parallel image processing algorithms for use on PASM are given.

PASM is to be a large-scale dynamically reconfigurable multimicroprocessor system. It is a special-purpose system aimed at exploiting the parallelism of image processing and pattern recognition tasks. PASM can be partitioned so that it operates as many independent SIMD and/or MIMD machines of various sizes, and it is being developed using a variety of problems in image processing and pattern recognition to guide the design choices. It can also be applied to related areas such as speech processing and biomedical signal processing.

PASM is to serve as a research tool for experimenting with parallel processing. The design attempts to incorporate the needed flexibility for studying large-scale SIMD and MIMD parallelism, while keeping system costs "reasonable." Portions of PASM have been simulated and a prototype is planned for the near future.

In section 2, the PASM organization is overviewed. Section 3 describes the Parallel Computation Unit. The Micro Controllers are discussed in section 4. In section 5, the secondary memory system is explored. Parallel algorithms for computing global histograms and 2-D FFTs are given in sections 6 and 7, respectively.

2. PASM Organization

A block diagram of the basic components of PASM is shown in Fig. 1. The Parallel Computation Unit (PCU) contains $N=2^n$ processors, N memory modules, and an interconnection network. The PCU processors are microprocessors that perform the actual SIMD and MIMD computations. The PCU memory modules are used by the PCU processors for data storage in SIMD mode and both data and instruction storage in MIMD mode. Thus, each PCU processor can operate in both the SIMD and MIMD modes of parallelism. The interconnection network provides a means of communication among the PCU processors and memory modules.

The Micro Controllers (MCs) are a set of microprocessors which act as the control units for the PCU processors in SIMD mode and orchestrate the activities of the PCU processors in MIMD mode. There are $Q=2^q$ MCs. Each MC controls N/Q PCU processors. A virtual SIMD

DTIC
COPY
INSPECTED
3

Accession For
NTIS GRA&I
DTIC TAB
Unannounced ☐
Justification
By
Distribution/
Availability Codes
Avail and/or
Dist Special
A-1

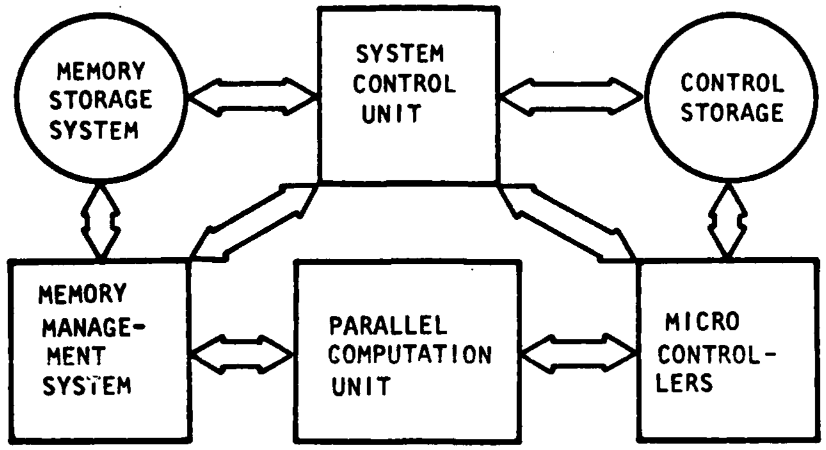


Fig. 1. Block diagram overview of PASM.

machine (partition) of size RN/Q, where $R=2^r$ and $1 \leq r \leq q$, is obtained by loading R MC memory modules with the same instructions simultaneously. Similarly, a virtual MIMD machine of size RN/Q is obtained by combining the efforts of the PCU processors of R MCs. Q is therefore the maximum number of partitions allowable, and N/Q is the size of the smallest partition. Possible values for N and Q are 1024 and 32, respectively. Control Storage contains the programs for the MCs.

The Memory Storage System provides secondary storage space for the data files in SIMD mode, and for the data and program files in MIMD mode. Multiple storage devices are used in the Memory Storage System to allow parallel data transfers. The Memory Management System controls the transferring of files between the Memory Storage System and the PCU memory modules. It employs a set of cooperating dedicated microprocessors.

The System Control Unit is a conventional machine, such as a PDP-11, and is responsible for the overall coordination of the activities of the other components of PASM. The types of tasks the System Control Unit will perform include program development, job scheduling, and coordination of the loading of the PCU memory modules from the Memory Storage System with the loading of the MC memory modules from Control Storage. By carefully choosing which tasks should be assigned to the System Control Unit and which should be assigned to other system components, the System Control Unit can work effectively and not become a bottleneck.

Sections 3 through 5 provide more information about the PASM system. References for further reading about PASM appear at the end of this paper.

3. Parallel Computation Unit

The Parallel Computation Unit (PCU) is shown in Fig. 2. A memory module is connected to each processor to form a processor - memory pair called a Processing Element (PE). The N PEs are numbered from 0 to N-1 and each PE knows its number (address). The interconnection network is used for communications among PEs. A pair of memory units is used for each memory module. This double-buffering scheme allows data to be moved between one memory unit and secondary storage (the Memory Storage System) while the processor operates on data in the other memory unit.

The PCU processors will be specially designed for parallel image processing. A PASM prototype (for N=16, Q=4) has been designed based on Motorola MC68000 processors. The final (N=1024) system would most likely employ custom VLSI processors.

Two types of multistage interconnection networks are being considered for PASM: the Generalized Cube [19] and the Augmented Data Manipulator (ADM) [18]. Features of the Generalized Cube network will be described to familiarize the readers with the properties of multistage networks.

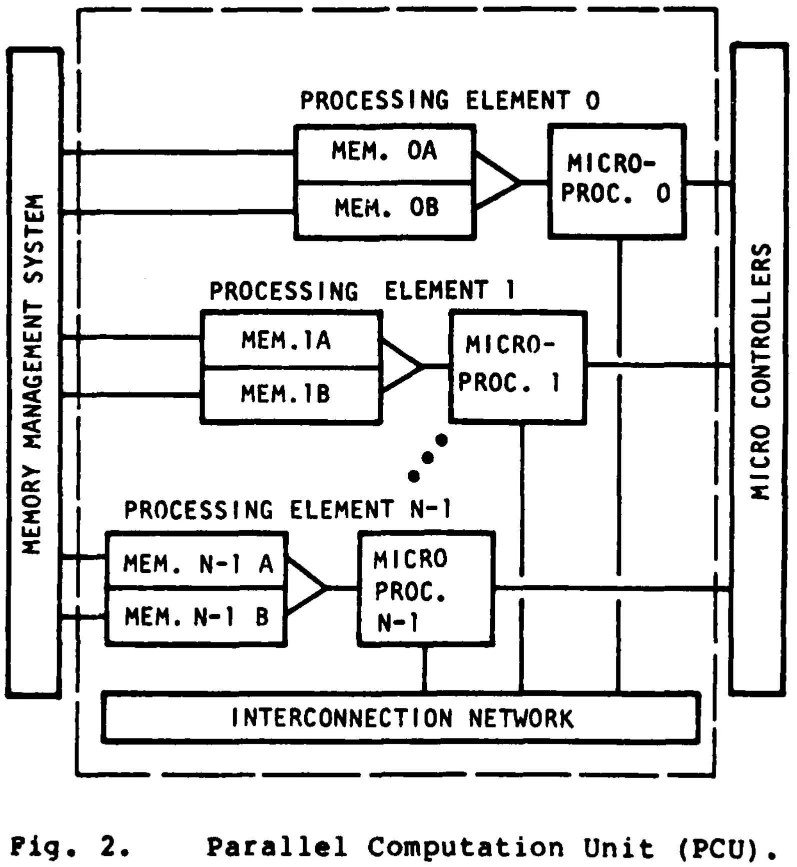


Fig. 2. Parallel Computation Unit (PCU).

The Generalized Cube network is a multistage cube-type network topology which was introduced as a standard for comparing network topologies. Other multistage cube-type networks include the baseline [26], delta [14], Extra Stage Cube [1], indirect binary n-cube [15], omega [12], STARAN flip [4], and SW-banyan (S=F=2) [10]. The Cube has N inputs and N outputs. It is shown in Fig. 3 for N=8. PE i, $0 \leq i < N$, would be connected to input port i and output port i of the unidirectional network.

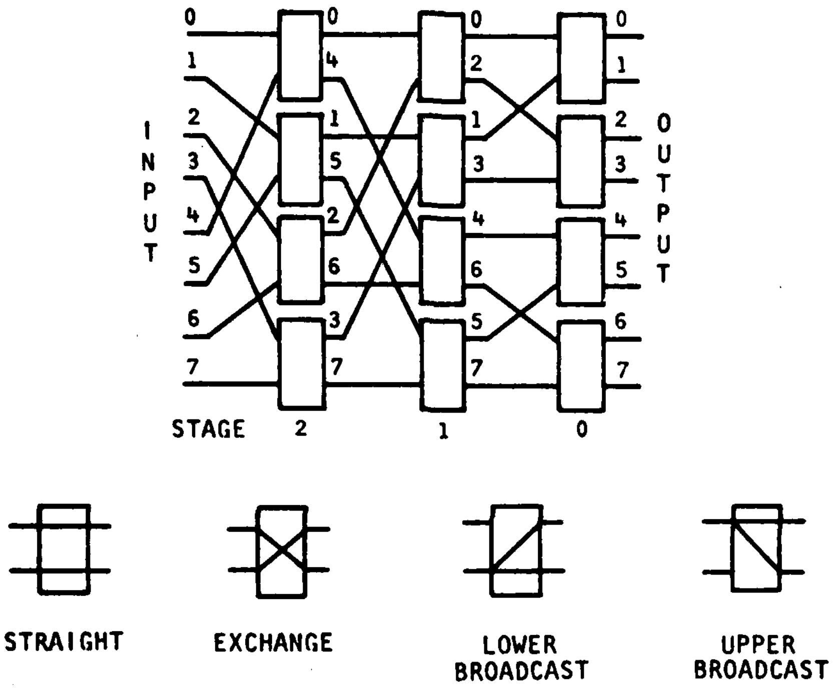


Fig. 3. Generalized Cube topology, shown for N=8.

The Generalized Cube topology has $n = \log_2 N$ stages, where each stage consists of a set of N lines connected to N/2 interchange boxes. Each interchange box is a two-input, two-output device. The labels of the input/output (I/O) lines entering the upper and lower inputs of an interchange box are used as the labels for the upper and lower outputs, respectively. Each interchange box can be set to one of the four legitimate states shown in Fig. 3.

The connections in this network are based on the cube interconnection functions [21, 22]. Let $P = p_{n-1} \cdots p_1 p_0$ be the binary representation of an arbitrary I/O line label. Then the n cube interconnection functions can be defined as:

$$\text{cube}_i(p_{n-1} \cdots p_1 p_0) = p_{n-1} \cdots p_{i+1} \overline{p}_i p_{i-1} \cdots p_1 p_0$$

where $0 \leq i < n$, $0 \leq P < N$, and $\overline{p}_i$ denotes the complement of p_i. This means that the cube_i interconnection function connects P to $\text{cube}_i(P)$, where $\text{cube}_i(P)$ is the I/O line whose label differs from P in just the i-th

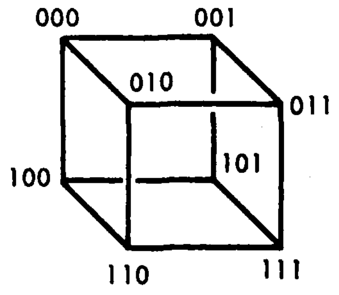


Fig. 4. Three-dimensional cube structure, with vertices labeled from 0 to 7 in binary.

bit position. Stage i of the Generalized Cube topology contains the cube_i interconnection function, i.e., it pairs I/O lines that differ only in the i-th bit position.

The reason that these interconnections are referred to as cube connections can be seen by considering the case for N=8. This is shown in Fig. 4. The eight vertices can be labeled so each vertex is connected to the n=3 vertices that differ from it in just one bit position. The horizontal connections are cube_0, the diagonals are cube_1, and the verticals are cube_2.

Using routing tags (as headers on messages) allows network control to be distributed among the PEs. The routing tags for one-to-one data transfers consist of n bits. If certain broadcast capabilities are included, then 2n bits are used. The routing tags set the state of each interchange box individually.

The n-bit routing tag for one-to-one connections is computed from the input port number and desired output port number. Let S be the source address (input port number) and D be the destination address (output port number). Then the routing tag $T = S \oplus D$ (where "$\oplus$" means bitwise "exclusive-or"). Let $t_{n-1} \ldots t_1 t_0$ be the binary representation of T. An interchange box in the network at stage i need only examine t_i. If $t_i=1$, an exchange is performed, and if $t_i=0$, the straight connection is used. For example, if N=8, S=011, and D=110, then T=101. The corresponding stage settings are exchange, straight, exchange. Because the exclusive-or operation is commutative, the incoming routing tag is the same as the return tag. Since the destination PE has the routing tag to the source PE, it is easy to perform handshaking if desired. The address of the source PE can be computed by the destination PE using $S = D \oplus T$.

Routing tags that can be used for broadcasting data are an extension of the above scheme. They are described in [19].

The Cube network can be partitioned into independent subnetworks of varying sizes. The <u>partitionability</u> of a network is its ability to divide the system into independent subsystems of different sizes. Furthermore, in this case, each subnetwork of size $N' \leq N$ will have all of the connection properties of a Cube network built to be of size N'.

The key to partitioning the Cube network so that each subnetwork is independent is based on the choice of the I/O ports that belong to the subnetworks. The requirement is that the addresses of all of the I/O ports in a partition of size 2^i agree (have the same values) in n-i of their bit positions.

For example, Fig. 5 shows one way a network of size eight can be partitioned into two subnetworks, each of size four. Group A consists of ports 0, 2, 4, and 6. Group B consists of ports 1, 3, 5, and 7. All ports in group A agree in the low-order bit position (it is a 0). All ports in group B agree in the low-order bit position (it is a 1). By setting all of the interchange boxes in stage 0 to straight, the two groups are isolated. This is because stage 0 is the only stage which allows input ports which differ in their low-order bit to exchange data. As stated above, each subnetwork has the properties of a Cube network. Thus, each subnetwork can be separately further subdivided, resulting in subnetworks of various sizes. This network property allows the PASM PCU PEs to be partitioned into independent virtual machines of various sizes.

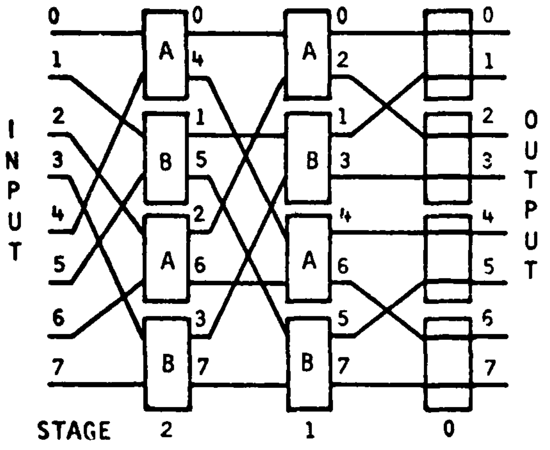


Fig. 5. Cube network of size eight partitioned into two subnetworks of size four based on the low-order bit position.

The routing tag scheme discussed previously can be used in conjunction with the partitioning concepts. Tags can be logically AND-ed with masks to force to 0 tag positions which correspond to interchange boxes which should be forced to the straight state.

The tradeoffs between the Cube and ADM multistage networks for PASM are currently under study. The ADM network is more flexible, but is more complex. The Cube may be more cost effective and sufficient for the system's needs. The Extra Stage Cube network [1] is a fault-tolerant variation of the Cube which is planned for inclusion in the PASM prototype.

In the following sections, it will be assumed that the PEs will be partitioned such that their addresses agree in the low-order bit positions. This constraint will allow either the Cube or ADM network to be used as the partitionable interconnection network in PASM.

4. Micro Controllers

In general, the possible advantages of a partitionable system include:

(a) fault tolerance - If a single PE fails, only those virtual machines (partitions) which must include the failed PE need to be disabled. The rest of the system can continue to function.

(b) multiple simultaneous users - Since there can be multiple independent virtual machines, there can be multiple simultaneous users of the system, each executing a different program.

(c) program development - Rather than trying to debug a program on, for example, 1024 PEs, it can be debugged on a smaller size virtual machine of 32 PEs.

(d) variable machine size for efficiency - If a task requires only N/2 of N available PEs, the other N/2 can be used for another task.

(e) subtask parallelism - Two independent subtasks that are part of the same job can be executed in parallel, sharing results if necessary.

Some form of multiple control units must be provided in order to have a partitionable SIMD/MIMD system. In PASM, this is done by having $Q=2^q$ MCs, physically addressed (numbered) from 0 to Q-1. Each MC controls N/Q PCU processors, as shown in Fig. 6.

Each MC is a microprocessor attached to a memory module. A memory module consists of a pair of memory units so that memory loading and computations can be overlapped. In SIMD mode, each MC fetches instructions from its memory module, executing the control flow instructions (e.g. branches) and broadcasting the data processing instruc-

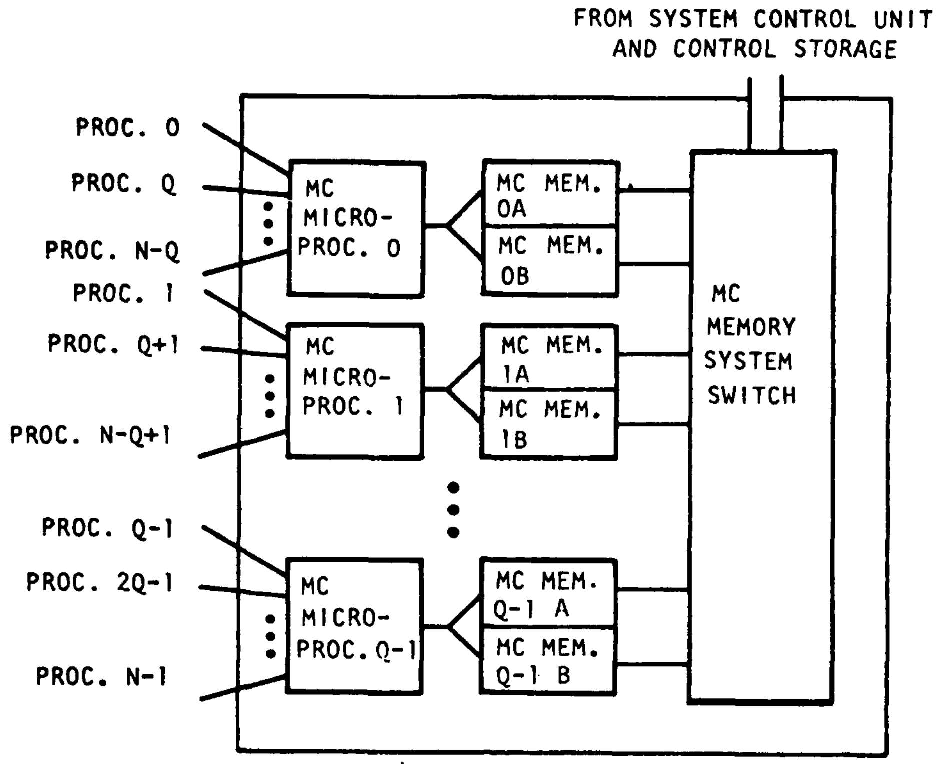


Fig. 6. PASM Micro Controllers (MCs).

tions to its PCU processors. The physical addresses of the N/Q processors which are connected to an MC must all have the same low-order q bits so that the network can be partitioned. The value of these low-order q bits is the physical address of the MC. A virtual SIMD machine of size RN/Q, where $R=2^r$ and $0 \leq r \leq q$, is obtained by loading R MCs with the same instructions and synchronizing the MCs. The physical addresses of these MCs must have the same low-order q-r bits so that all of the PCU processors in the partition have the same low-order q-r physical address bits. Similarly, a virtual MIMD machine of size RN/Q is obtained by combining the efforts of the PCU PEs associated with R MCs which have the same low-order q-r physical address bits. In MIMD mode, the MCs may be used to help coordinate the activities of their PCU PEs.

Permanently assigning a fixed number of PCU PEs to each MC has several advantages over allowing a varying assignment, such as used in MAP. One advantage is that the operating system need only schedule (and monitor the "busy" status of) Q MCs, rather than N PCU PEs. When Q=32 and N=1024, this is a substantial savings. Another advantage is that no crossbar switch is needed for connecting processors and control units (such as proposed for MAP [13]). A third advantage is that it supports network partitioning. In addition, this fixed connection scheme allows the efficient use of multiple secondary storage devices, which is discussed below. The main disadvantage of this approach is that each virtual machine size must be a power of two, with a minimum value of N/Q. However, for PASM's intended experimental environment, flexibility at reasonable cost is the goal, not maximum processor utilization.

The loading of programs from Control Storage into the MC memory units is controlled by the System Control Unit. When large SIMD jobs are run, that is, jobs which require more than N/Q processors, more than one MC executes the same set of instructions. Each MC has its own memory, so that if more than one MC is to be used, several memories must be loaded with the same set of instructions. The fastest way to load several MC memories with the same set of instructions is to load all of the memories at the same time. A shared bus from Control Storage is used to do this parallel loading.

This basic MC organization can be enhanced to allow the sharing of memory modules by the MCs in a partition. The MCs can be connected by a shared reconfigurable ("shortable") bus [2, 11], as shown in Fig. 7. The MCs must be ordered on the bus in terms of the bit reverse of their addresses due to the partitioning rules. This enhanced MC connection scheme could provide more program space for jobs using multi-

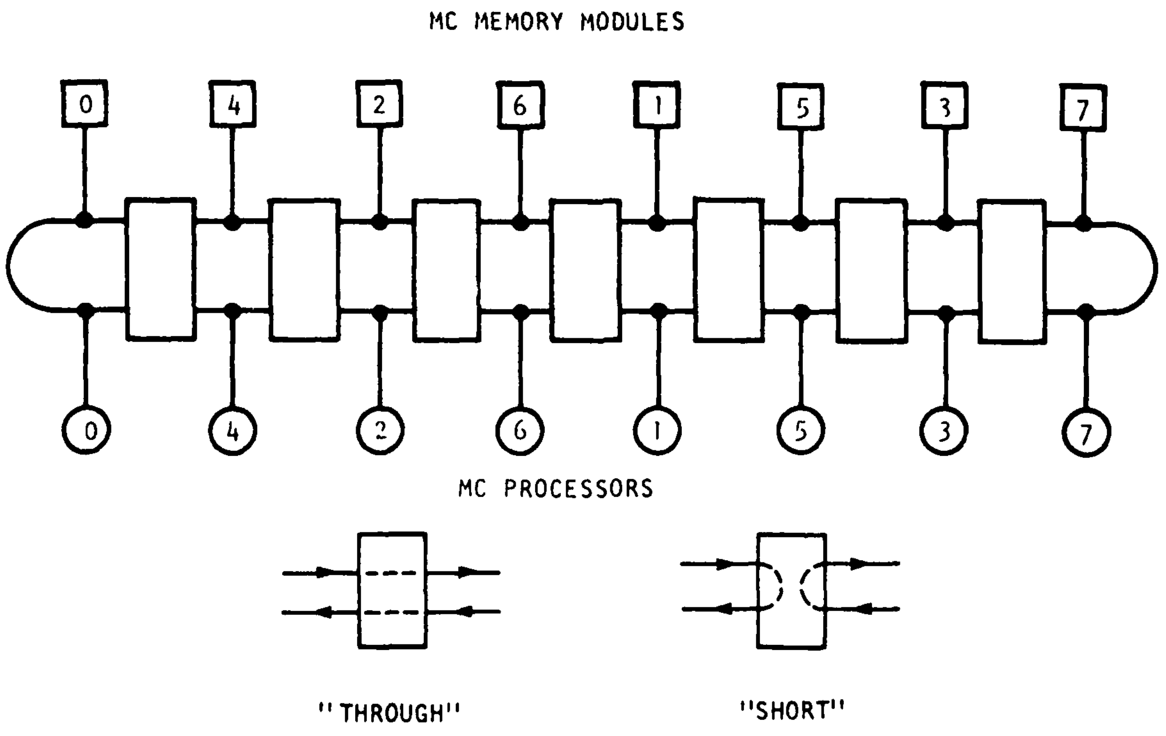


Fig. 7. Reconfigurable shared bus scneme for interconnecting MC processors and MC memory modules, shown for Q=8. Each box can be set to "through" or "short."

ple MCs and would also provide a degree of fault tolerance, since known-faulty MC memory modules could be ignored. These advantages come at the expense of additional system complexity, and the inclusion of the enhanced scheme in PASM will depend on cost constraints at implementation time.

Within each partition the PCU processors and memory modules are assigned logical addresses. Given a virtual machine of size RN/Q, the processors and memory modules for this partition have logical addresses (numbers) 0 to (RN/Q)-1, $R=2^r$, $0 \le r \le q$. The logical number of a PCU PE is the high-order r+n-q bits of its physical number. Similarly, the MCs assigned to the partition are logically numbered (addressed) from 0 to R-1. For R>1, the logical number of an MC is the high-order r bits of its physical number. The PASM language compilers and operating system will be used to convert from logical to physical addresses, so a system user will deal only with logical addresses.

There are instructions which examine the collective status of all of the PEs of a virtual SIMD machine, such as "if any," "if all," and "if none." These instructions change the flow of control of the program at execution time depending on whether any or all processors in the virtual SIMD machine satisfy some condition. For example, if each PE is processing data from a different section of a radar unit, but all PEs are looking for enemy planes, it is desirable to know "if any" of the PEs has discovered a possible attack. This requires communication among the MCs comprising the virtual SIMD machine. There is a set of buses shared by MCs for this purpose.

When operating in SIMD mode, all of the active PCU PEs will execute instructions broadcast to them by their MC. A masking scheme is a method for determining which PCU PEs will be active at a given point in time. PASM will use PE address masks and data conditional masks.

The PE address masking scheme uses an n-position mask to specify which of the N PCU PEs are to be activated. Each position of the mask corresponds to a bit position in the addresses of the PEs. Each position of the mask will contain either a 0, 1, or X ("don't care") and the only PEs that will be active are those whose address matches the mask: 0 matches 0, 1 matches 1, and either 0 or 1 matches X. Square brackets denote a mask. Superscripts are used as repetition factors. For example: MASK $[X^{n-1}1]$ activates all odd-numbered PEs; MASK $[1^{n-i}X^i]$ activates PEs $N-2^i$ to N-1. PE address masks are specified in the SIMD program.

A negative PE address mask is similar to a regular PE address mask, except that it activates all those PEs which do not match the mask.

Negative PE address masks are prefixed with a minus sign to distinguish them from regular PE address masks. For example, for N=8, MASK [-01X] activates all PEs except 2 and 3. This type of mask can activate sets of PEs a single regular PE address mask cannot.

Data conditional masks will be implemented in PASM for use when the decision to enable and disable PEs is made at execution time. Data conditional masks are the implicit result of performing a conditional branch dependent on local data in an SIMD machine environment, where the result of different PEs' evaluations may differ. As a result of a conditional where statement of the form

where <data-condition> do ... elsewhere ...

each PE will set its own flag to activate itself for either the "do" or the "elsewhere," but not both. The execution of the "elsewhere" statements must follow the "do" statements; i.e., the "do" and "elsewhere" statements cannot be executed simultaneously. For example, as a result of executing the statement:

where $A < B$ do $C \leftarrow A$ elsewhere $C \leftarrow B$

each PE will load its C register with the minimum of its A and B registers, i.e., some PEs will execute "$C \leftarrow A$," and then the rest will execute "$C \leftarrow B$." This type of masking is used in such machines as the Illiac IV [3] and PEPE [7]. "Where" statements can be nested using a run-time control stack.

5. Secondary Memory System

The Memory Storage System will consist of N/Q independent Memory Storage Units, numbered from 0 to (N/Q)-1. These devices will allow fast loading and unloading of the N double-buffered PCU memory modules and will provide storage for system image data and MIMD programs.

Each Memory Storage Unit is connected to Q PCU memory modules. For $0 \leq i < N/Q$, Memory Storage Unit i is connected to those memory modules whose physical addresses are of the form $(Q*i)+k$, $0 \leq k < Q$. Recall that, for $0 \leq k < Q$, MC k is connected to those PEs whose physical addresses are of the form $(Q*i)+k$, $0 \leq i < N/Q$. This is shown for N=32 and Q=4 in Fig. 8.

For a partition of size N/Q, the two main advantages of this approach are that (1) all of the memory modules can be loaded in parallel and (2) the data is directly available no matter which partition (MC group) is chosen. This is done by storing in Memory Storage Unit

A virtual machine of RN/Q PEs, $1 \leq R \leq Q$, logically numbered from 0 to RN/Q-1, requires only R parallel block loads if the data for the memory module whose high-order n-q logical address bits equal i is loaded into Memory Storage Unit i. This is true no matter which group of R MCs (which agree in their low-order q-r address bits) is chosen.

As an example, consider Fig. 8, and assume a virtual machine of size 16 is desired. The data for the memory modules whose logical addresses are 0 and 1 is loaded into Memory Storage Unit 0, for memory modules 2 and 3 into unit 1, etc. Assume the partition of size 16 is chosen to consist of the processors connected to MCs 1 and 3. Given this assignment of MCs, the PCU memory module whose physical address is 2*i+1 has logical address i, $0 \leq i < 16$. The Memory Storage Units first load memory modules physically addressed 1, 5, 9, 13, 17, 21, 25, and 29 (simultaneously), and then load memory modules 3, 7, 11, 15, 19, 23, 27, and 31 (simultaneously). No matter which pair of MCs is chosen, only two parallel block loads are needed. Thus, for a virtual machine of size RN/Q, this secondary storage scheme allows all RN/Q memory modules to be loaded in R parallel block transfers, $1 \leq R \leq Q$.

This same approach can be taken if only $(N/Q)/2^d$ distinct Memory Storage Units are available, where $0 \leq d \leq n-q$. In this case, however, $R2^d$ parallel block loads will be required instead of just R. The number and types of devices that will be used in PASM will depend upon speed requirements, cost constraints, and the state-of-the-art of storage technology at implementation time.

The <u>Memory Management System</u> is composed of a separate set of microprocessors dedicated to performing tasks in a distributed fashion, i.e., one processor handles Memory Storage System bus control, one handles the peripheral device I/O, etc. This distributed processing approach is chosen in order to provide the Memory Management System with a large amount of processing power at low cost. The division of tasks chosen is based on the main functions which the Memory Management System must perform, including: (1) generating tasks based on PCU memory module load/unload requests from the System Control Unit; (2) scheduling of Memory Storage System data transfers; (3) control of input/output operations involving peripheral devices and the Memory Storage System; (4) maintenance of the Memory Management System file directory information; and (5) control of the Memory Storage System bus system.

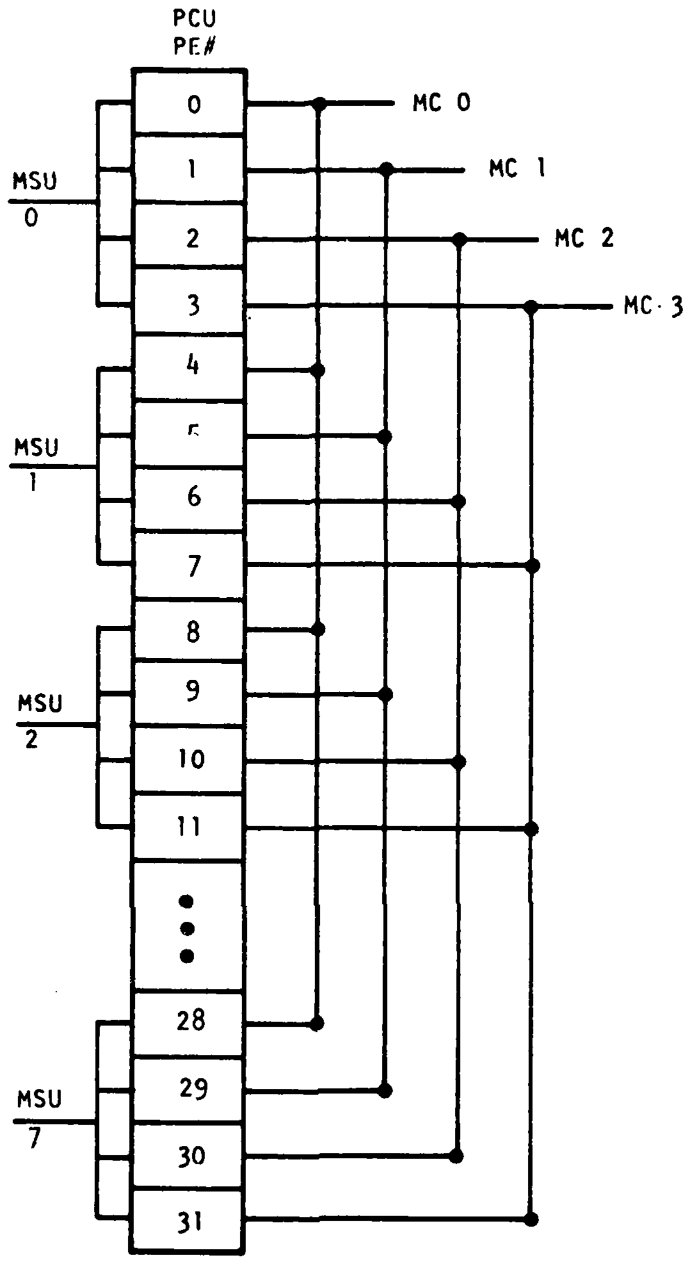


Fig. 8. Organization of the Memory Storage System, shown for N=32 and Q=4. "MSU" is Memory Storage Unit.

i the data for a task which is to be loaded into the i-th logical memory module of the virtual machine of size N/Q, $0 \leq i < N/Q$. Memory Storage Unit i is connected to the i-th memory module in each MC group so that no matter which MC group of N/Q processors is chosen, the data from the i-th Memory Storage unit can be loaded into the i-th logical memory module, $0 \leq i < N/Q$, simultaneously. Thus, for virtual machines of size N/Q, this secondary storage scheme allows all N/Q memory modules to be loaded in one parallel block transfer.

6. Parallel Computation of a Global Histogram

In this section, an SIMD algorithm for computing the global histogram of an algorithm is given [20]. Assume there are $\underline{B}=2^b$ bins in the histogram, $B \le N$. An M by M image is represented by an array of M^2 pixels (picture elements), where the value of each pixel is assumed to be a b-bit unsigned integer representing one of B possible gray levels. The B-bin histogram of the image contains a j in bin i if exactly j of the pixels have a gray level of i, $0 \le i < B$.

Assume the image is equally distributed among the N PEs in PASM, i.e., each PE has M^2/N pixels, and $B \le M^2/N$. Since the image is distributed over N PEs, each PE will calculate a B-bin histogram based on its M^2/N segment of the image. Then these "local" histograms will be combined using the algorithm described below. This algorithm is demonstrated for N=16 and B=4 bins in Fig. 9.

Each block of B PEs performs B simultaneous recursive doublings [24] to compute the histogram for the portion of the image contained in the block in the first b steps. At the end of the b steps, each PE has one bin of this partial histogram. This is accomplished by first dividing the B PEs of a block into two groups. Each group accumulates

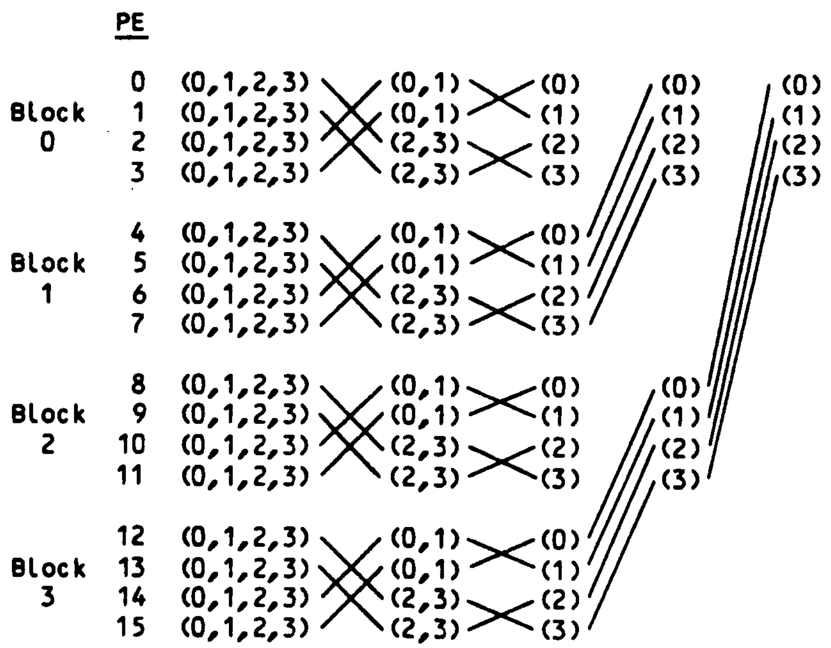


Fig. 9. Histogram calculation for N=16 PEs, B=4 bins. (w,...,z) denotes that bins w through z of the partial histogram are in the PE.

the sums for half of the bins, and sends the bins it is not accumulating to the group which is accumulating those bins. At each step of the algorithm, each group of PEs is divided in half such that the PEs with the lower addresses form one group, and the PEs with the higher addresses form another. The accumulated sums are similarly divided in half based on their indices in the histogram. The groups then exchange sums, so that each PE contains only sum terms which it is accumulating. The newly-received sums are added to the sums already in the PE. After b steps, each PE has the total value for one bin from the portion of the image contained in the B PEs in its block.

The results for these blocks can be combined in n-b steps to yield the histogram of the entire image distributed over B PEs, with the sum for bin i in PE i, $0 \leq i < B$. This is done by performing n-b steps of a recursive doubling [24] algorithm to sum the partial histograms from the N/B blocks, shown by the last two steps of Fig. 9. Note that B recursive doublings are being performed simultaneously, one for each bin. A general algorithm to compute the B-bin histogram for an image distributed over N PEs is given in [20].

Now consider relative speeds of sequential and parallel computation of the histogram. A sequential algorithm to compute the histogram of an M by M image requires M^2 additions. The SIMD algorithm uses M^2/N additions for each PE to compute its local histogram. At step i in the merging of the partial histograms, $0 \leq i < b$, the number of parallel data transfer/adds required is $B/2^{i+1}$. A total of B-1 transfer/adds are therefore performed in the first b steps of the algorithm. Then n-b parallel transfers and additions are needed to combine the block histograms. This technique therefore requires B-1+n-b parallel transfer/add operations, plus the M^2/N additions needed to compute the local PE histograms. For example, if N=1024, M=512, and B=128, the sequential algorithm would require 262,144 additions; the parallel algorithm uses 256 addition steps plus 130 transfer/add steps. The result of the algorithm, i.e., the histogram, is distributed over the first B PEs. This distribution may be efficient for further processing on the histogram, e.g., finding the maximum or minimum, or for smoothing the histogram. If it is necessary for the entire histogram to be in a single PE, B-1 additional parallel data transfers are required. Both the Cube and ADM multistage networks can perform all of the required inter-PE data transfers efficiently.

7. 2-D FFT Algorithms

In this section, an SIMD algorithm to compute the 2-D FFT of an image is given [23]. A standard approach to computing the 2D-DFT of an image S is to perform the 1-D DFT on the rows of S, giving an intermediate matrix G, and then perform the 1-D DFT on the columns of G. The resulting matrix F is the 2-D DFT of S. Suppose that an SIMD machine has N=M PEs, each of which has one row of an M by M input image S. An efficient method for obtaining F, the DFT of S, is to perform M 1-D FFTs in parallel on the rows of S to get G, "transpose" G, and then perform M 1-D FFTs in parallel on the columns of G to get F^T. This is shown in Fig. 10. (F^T can be transposed to give F, however, this may not be necessary depending on what further processing is done on F.)

To form the transpose of G, G^T, such that each row of G^T is in a different PE, the basic operation performed is the transfer of array

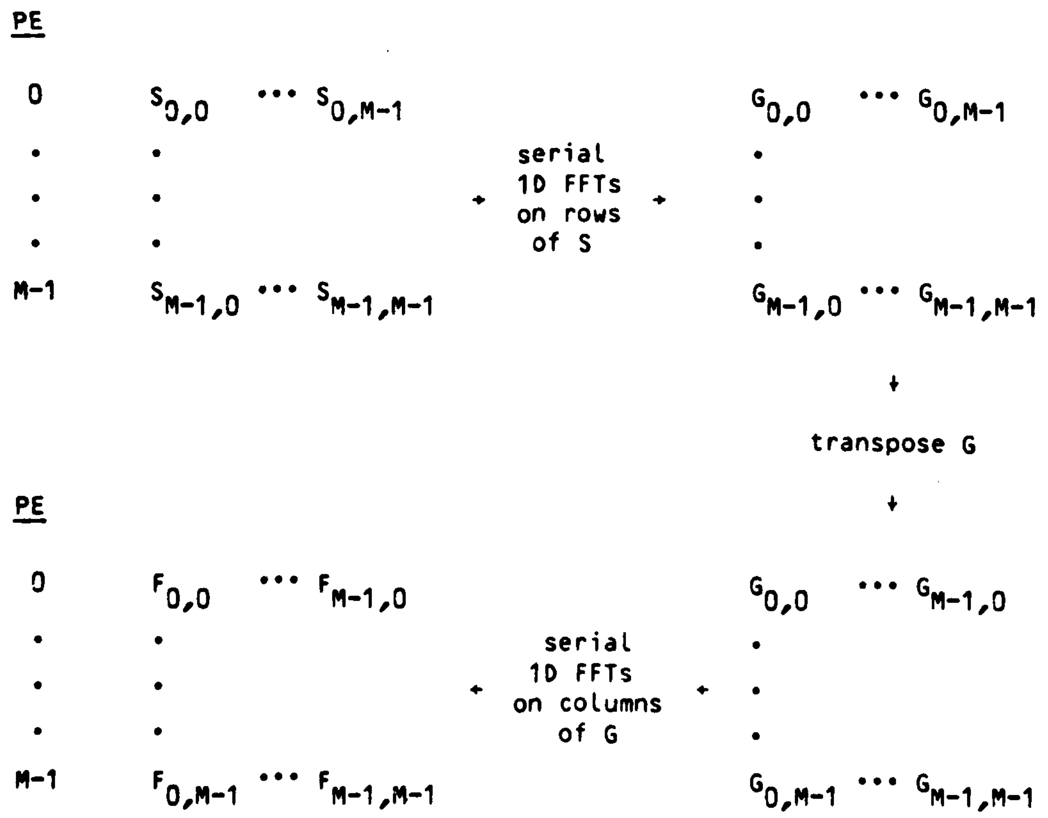

PE
0
M-1
$S_{0,0}$ ... $S_{0,M-1}$
$S_{M-1,0}$ ... $S_{M-1,M-1}$
serial 1D FFTs on rows of S
$G_{0,0}$ ... $G_{0,M-1}$
$G_{M-1,0}$ ... $G_{M-1,M-1}$
transpose G
PE
0
M-1
$F_{0,0}$ ... $F_{M-1,0}$
$F_{0,M-1}$ ... $F_{M-1,M-1}$
serial 1D FFTs on columns of G
$G_{0,0}$ ... $G_{M-1,0}$
$G_{0,M-1}$ ... $G_{M-1,M-1}$

Fig. 10. Computation of 2-D FFT of M by M array S using M PEs.

element G(v,w) from PE v to PE w. This is done for M G(v,w)'s in parallel by sending data from PE v to PE (v+i) mod M for all of the G(v,w) for which (w-v) mod M = i. The parallel transfer operation is performed for $1 \leq i < M$. For each i value, the element which PE v sends is the w-th element of the row of G held in PE v, where w = (v+i) mod M. That element, received in PE w, is stored as the v-th element of the column of G being created in PE w, where v = (w-i) mod M. The elements on the diagonal G(v,w), where v=w, do not have to be transferred. Performing the transpose therefore requires M-1 parallel data transfers.

The serial complexity of 2M 1-D FFTs (i.e., an M by M 2-D DFT) is $M^2\log_2 M$ "butterflies." The above parallel implementation of the 2-D DFT executes two serial FFT algorithms and has a complexity of $M\log_2 M$ butterfly steps. Thus, an ideal speedup of M is achieved for butterfly operations with a cost of M-1 data transfers.

This approach can be generalized for N<M. For example, if N=M/2 each PE is given two rows of the input matrix S. The FFTs on the rows of S are performed by two serial FFTs, executed one after the other, on the two rows in each PE. This yields G, with each PE having two rows of G. The second step is to form the transpose of G, G^T, where each PE has two rows of G^T (i.e., each PE has two columns of G). If PE i contains rows 2i and 2i+1, then, in general, G(i,j) is transferred from PE $\lfloor i/2 \rfloor$ to PE $\lfloor j/2 \rfloor$, $0 \leq i,j < M$. The complexity associated with the transpose is 2M-4 parallel transfers. The -4 term appears because the diagonal and near-diagonal terms are already in the correct PE. The final step is to perform a 1-D DFT on the columns of G. This is done by two serial FFTs in each PE, as above. This gives F^T, with each PE having two rows of F^T. This implementation has a complexity of four serial FFT algorithms, or $2M\log_2 M$ butterfly steps. This is the maximum possible reduction in the number of butterfly steps, given M/2 PEs. The overhead associated with the transpose is 2M-4 transfers.

In general, when this method is implemented on N PEs, $N \leq M$, the complexity will be derived directly from the 1-D FFT algorithm used. If the complexity of the serial 1-D FFT algorithm is C, then the complexity of the 2-D FFT algorithm is 2(M/N)C plus the cost of computing the transpose. If $N = M/(2^r)$, the cost of the transpose is $2^r(M-2^r)$ data transfers. The -2^r term appears because before the transpose each PE holds 2^r rows, and after the transpose each PE holds 2^r columns. Thus, only $M-2^r$ elements of each row need to be transferred. In all cases, the necessary inter-PE data transfers can be done efficiently by the Cube and ADM multistage networks.

Table 1. The PASM design parameters, based on current plans.

	general	full PASM	PASM prototype
Number of PEs	N	1024	16
Number of network stages (Extra Stage Cube)	$\log_2 N + 1$	11	5
Number of MCs	Q	32	4
Number of PEs per MC	N/Q	32	4
Number of Memory Storage Units	N/Q	32	4
Number of Memory Management System processors	fixed	5	5
Smallest size partition	N/Q	32	4
Maximum number of partitions	Q	32	4

8. Conclusions

This paper provided an overview of the PASM system and examples of its use. Table 1 summarizes the PASM design parameters. In order to contrast PASM to a different approach to parallel image processing, Table 2 compares the features of CLIP4 [8] to the planned features of PASM. A reading list for further information about PASM is provided at the end of this paper.

Table 2. A comparison of the features of CLIP4 and the planned features of PASM.

feature	CLIP4	PASM
Year built	1980	1983/4 ? (prototype)
Processor type	1-bit, simple	32-bit, complex (68000 prototype)
Memory size per processor	32 bits	64K words
Network type	8 nearest neighbors	multistage
Number of processors for computation	$96^2 = 9K$	1024 (16 prototype)
Image division	pixel/processor	subimage/PE
I/O	shift by column, rows in parallel	double-buffered PE memories, multiple secondary storage devices
Modes	SIMD	partitionable SIMD/MIMD

In conclusion, the objective of the PASM design is to achieve a system which attains a compromise between flexibility and cost-effectiveness for a specific problem domain. A dynamically reconfigurable system such as PASM should be a valuable tool for both image processing/pattern recognition and parallel processing research.

Acknowledgements

Portions of this overview of the PASM system and its use in parallel image processing are based on [19], [20], and [23]. The contributions of my co-authors on these papers are gratefully acknowledged. In addition, James T. Kuehn provided valuable comments and aid. I thank Mickey Krebs for typing the manuscript.

References

[1] G. B. Adams III and H. J. Siegel, "The extra stage cube: A fault-tolerant interconnection network for supersystems," IEEE Trans. Computers, Vol. C-31, May 1982, pp. 443-454.

[2] R. Arnold and E. Page, "A hierarchical, restructurable multimicroprocessor architecture," 3rd Symp. Computer Architecture, Jan. 1976, pp. 40-45.

[3] G. Barnes, et al., "The Illiac IV computer," IEEE Trans. Computers, Vol. C-17, Aug. 1968, pp. 746-757.

[4] K. E. Batcher, "The flip network in STARAN," 1976 Int'l. Conf. Parallel Processing, Aug. 1976, pp. 65-71.

[5] K. E. Batcher, "STARAN series E," 1977 Int'l. Conf. Parallel Processing, Aug. 1977, pp. 144-153.

[6] W. J. Bouknight, et al., "The Illiac IV system," Proc. IEEE, Vol. 60, Apr. 1972, pp. 369-388.

[7] B. A. Crane, et al., "PEPE computer architecture," COMPCON 1972, Sept. 1972, pp. 57-60.

[8] M. J. B. Duff, "Architectures of SIMD cellular logic image processing arrays," this volume.

[9] M. J. Flynn, "Very high-speed computing systems," Proc. IEEE, . . 4, Dec. 1966, pp. 1901-1909.

[10] L. R. Goke and G. J. Lipovski, "Banyan networks for partitioning multimicroprocessor systems," 1st Symp. Computer Architecture, Dec. 1973, pp. 21-28.

[11] S. I. Kartashev and S. P. Kartashev, "A multicomputer system with dynamic architecture," IEEE Trans. Computers, Vol. C-28, Oct. 1979, pp. 704-720.

[12] D. H. Lawrie, "Access and alignment of data in an array processor," IEEE Trans. Computers, Vol. C-24, Dec. 1975, pp. 1145-1155.

[13] G. J. Nutt, "Microprocessor implementation of a parallel processor," 4th Symp. Computer Architecture, Mar. 1977, pp. 147-152.

[14] J. H. Patel, "Performance of processor-memory interconnections for multiprocessors," IEEE Trans. Computers, Vol. C-30, Oct. 1981, pp. 771-780.

[15] M. C. Pease, III, "The indirect binary n-cube microprocessor array," IEEE Trans. Computers, Vol. C-26, May 1977, pp. 458-473.

[16] J. L. Potter, "MPP architecture and programming," in Multicomputers and Image Processing: Algorithms and Programs, K. Preston and L. Uhr, eds., Academic Press, New York, NY, 1982, pp. 275-290.

[17] M. C. Sejnowski, E. T. Upchurch, R. N. Kapur, D. P. S. Charlu, and G. J. Lipovski, "An overview of the Texas Reconfigurable Array Computer," AFIPS 1980 Nat'l. Computer Conf., June 1980, pp. 631-641.

[18] H. J. Siegel and R. J. McMillen, "Using the augmented data manipulator network in PASM," Computer, Vol. 14, Feb. 1981, pp. 25-33.

[19] H. J. Siegel and R. J. McMillen, "The multistage cube: a versatile interconnection network," Computer, Vol. 14, Dec. 1981, pp. 65-76.

[20] H. J. Siegel, L. J. Siegel, F. C. Kemmerer, P. T. Mueller, Jr., H. E. Smalley, and S. D. Smith, "PASM: a partitionable SIMD/MIMD system for image processing and pattern recognition," IEEE Trans. Computers, Vol. C-30, Dec. 1981, pp. 934-947.

[21] H. J. Siegel, "Analysis techniques for SIMD machine interconnection networks and the effects of processor address masks," IEEE Trans. Computers, Vol. C-26, Feb. 1977, pp. 153-161.

[22] H. J. Siegel, "A model of SIMD machines and a comparison of various interconnection networks," IEEE Trans. Computers, Vol. C-28, Dec. 1979, pp. 907-917.

[23] L. J. Siegel, P. T. Mueller, Jr., and H. J. Siegel, "FFT algorithms for SIMD machines," 17th Allerton Conf. Communication, Control, and Computing, Oct. 1979, pp. 1006-1015.

[24] H. S. Stone, "Parallel computers," in Introduction to Computer Architecture, 2nd edition, edited by H. S. Stone, Science Research Associates, Inc., Chicago, IL, 1980, pp. 363-425.

[25] R. J. Swan, S. H. Fuller, and D. P. Siewiorek, "Cm*: a modular, multi-microprocessor," Nat'l. Computer Conf., June 1977, pp. 637-644.

[26] C. L. Wu and T. Y. Feng, "On a class of multistage interconnection networks," IEEE Trans. Computers, Vol. C-29, Aug. 1980, pp. 694-702.

[27] W. A. Wulf and C. G. Bell, "C.mmp - a multi-miniprocessor," Fall Joint Computer Conf., Dec. 1972, pp. 765-777.

Further reading about PASM

reconfigurable organization:

H. J. Siegel, P. T. Mueller, Jr., and H. E. Smalley, Jr., "Control of a partitionable multimicroprocessor system," 1978 Int'l. Conf. Parallel Processing, Aug. 1978, pp. 9-17.

H. J. Siegel, L. J. Siegel, F. C. Kemmerer, P. T. Mueller, Jr., H. E. Smalley, and S. D. Smith, "PASM: a partitionable SIMD/MIMD system for image processing and pattern recognition," IEEE Trans. Computers, Vol. C-30, Dec. 1981, pp. 934-947.

parallel memory management system:

H. J. Siegel, F. Kemmerer, and M. Washburn, "Parallel memory system for a partitionable SIMD/MIMD machine," 1979 Int'l. Conf. Parallel Processing, Aug. 1979, pp. 212-221.

J. T. Kuehn, H. J. Siegel, and M. Grosz, "A distributed memory management system for PASM," IEEE Comp. Soc. Workshop Computer Architecture for Pattern Analysis and Image Database Management, Oct. 1983.

interconnection network - multistage cube:

R. J. McMillen and H. J. Siegel, "The hybrid cube network," Distributed Data Acquisition, Computing, and Control Symp., Dec. 1980, pp. 11-22.

R. J. McMillen, G. B. Adams III, and H. J. Siegel, "Performance and implementation of 4x4 switching nodes in an interconnection network for PASM," 1981 Int'l. Conf. Parallel Processing, Aug. 1981, pp. 229-233.

H. J. Siegel and R. J. McMillen, "The multistage cube: a versatile interconnection network," Computer, Vol. 14, Dec. 1981, pp. 65-76.

G. B. Adams III and H. J. Siegel, "The extra stage cube: a fault-tolerant interconnection network for supersystems," IEEE Trans. Computers, Vol. C-31, May 1982, pp. 443-454.

interconnection network - ADM:

S. D. Smith, H. J Siegel, R. J. McMillen, and G. B. Adams III, "Use of the augmented data manipulator multistage network for SIMD machines," 1980 Int'l. Conf. Parallel Processing, Aug. 1980, pp. 75-78.

R. J. McMillen, G. B. Adams III, and H. J. Siegel, "Permuting with the augmented data manipulator network," 18th Allerton Conf. Communication, Control, and Computing, Oct. 1980, pp. 544-553.

H. J. Siegel and R. J. McMillen, "Using the augmented data manipulator network in PASM," Computer, Vol. 14, Feb. 1981, pp. 25-33.

R. J. McMillen and H. J. Siegel, "Performance and fault tolerance improvements in the inverse augmented data manipulator network," 9th Int'l. Symp. Computer Architecture, Apr. 1982, pp. 63-72.

G. B. Adams III and H. J. Siegel, "On the number of permutations performable by the augmented data manipulator network," IEEE Trans. Computers, Vol. C-31, Apr. 1982, pp. 270-277.

R. J. McMillen and H. J. Siegel, "Routing schemes for the augmented data manipulator network in an MIMD system," IEEE Trans. Computers, Dec. 1982, pp. 63-72.

interconnection network - comparisons:

H. J. Siegel and S. D. Smith, "Study of multistage SIMD interconnection networks," 5th Symp. Computer Architecture, Apr. 1978, pp. 223-229.

H. J. Siegel, "Interconnection networks for SIMD machines," Computer, Vol. 12, June 1979, pp. 57-65.

H. J. Siegel, R. J. McMillen, and P. T. Mueller, Jr., "A survey of interconnection methods for reconfigurable parallel processing systems," 1979 Nat'l. Computer Conf., June 1979, pp. 529-542.

H. J. Siegel, "The theory underlying the partitioning of permutation networks," IEEE Trans. Computers, Vol. C-29, Sept. 1980, pp. 791-801.

R. J. McMillen and H. J. Siegel, "A comparison of cube type and data manipulator type networks," 3rd Int'l. Conf. Distributed Computing Systems, Oct. 1982, pp. 614-621.

H. J. Siegel, Interconnection Networks for Large-Scale Parallel Processing: Theory and Case Studies, D.C. Heath and Co., Lexington, MA, 1983.

distributed operating system:

H. J. Siegel, L. J. Siegel, R. J. McMillen, P. T. Mueller, Jr., and S. D. Smith, "An SIMD/MIMD multimicroprocessor system for image processing and pattern recognition," 1979 IEEE Comp. Soc. Conf. Pattern Recognition and Image Processing, Aug. 1979, pp. 214-224.

D. L. Tuomenoksa and H. J. Siegel, "Application of two-dimensional bin packing algorithms for task scheduling in the PASM multimicrocomputer system," 19th Allerton Conf. Communication, Control, and [illegible]

D. L. Tuomenoksa and H. J. Siegel, "Analysis of the PASM control system memory hierarchy," *1982 Int'l. Conf. Parallel Processing*, Aug. 1982, pp. 363-370.

D. L. Tuomenoksa and H. J. Siegel, "Analysis of multiple-queue task scheduling algorithms for multiple-SIMD machines," *3rd Int'l. Conf. Distributed Computing Systems*, Oct. 1982, pp. 114-121.

D. L. Tuomenoksa and H. J. Siegel, "Preloading schemes for the PASM parallel memory system," *1983 Int'l. Conf. Parallel Processing*, Aug. 1983, pp. 407-415.

prototype design:

J. T. Kuehn and H. J. Siegel, "Simulation studies of PASM in SIMD mode," *1981 IEEE Comp. Soc. Workshop Computer Architecture for Pattern Analysis and Image Database Management*, Nov. 1981, pp. 43-50.

J. T. Kuehn, H. J. Siegel, and P. D. Hallenbeck, "Design and simulation of an MC68000-based multimicroprocessor system," *1982 Int'l. Conf. Parallel Processing*, Aug. 1982, pp. 353-362.

parallel programming language:

P. T. Mueller, Jr., L. J. Siegel, and H. J. Siegel, "A parallel language for image and speech processing," *IEEE Comp. Soc. Fourth Int'l. Computer Software and Applications Conference (COMPSAC 80)*, Oct. 1980, pp. 476-483.

C. Cline and H. J. Siegel, "Extensions of Ada for SIMD parallel processing," *IEEE Comp. Soc. Seventh Int'l. Computer Software and Applications Conf. (COMPSAC 83)*, Nov. 1983.

parallel image processing:

L. J. Siegel, P. T. Mueller, Jr., and H. J. Siegel, "FFT algorithms for SIMD machines," *17th Allerton Conf. Communication, Control, and Computing*, Oct. 1979, pp. 1006-1015.

P. T. Mueller, Jr., L. J. Siegel, and H. J. Siegel, "Parallel algorithms for the two-dimensional FFT," *5th Int'l. Conf. Pattern Recognition*, Dec. 1980, pp. 497-502.

P. H. Swain, H. J. Siegel, and J. El-Achkar, "Multiprocessor implementation of image pattern recognition: a general approach," *5th Int'l. Conf. Pattern Recognition*, Dec. 1980, pp. 309-317.

H. J. Siegel and P. H. Swain, "Contextual classification on PASM," *IEEE Comp. Soc. Conf. Pattern Recognition and Image Processing*, Aug. 1981, pp. 320-325.

T. N. Mudge, E. J. Delp, L. J. Siegel, and H. J. Siegel, "Image coding using the multimicroprocessor system PASM," *IEEE Comp. Soc. Conf. Pattern Recognition and Image Processing*, June 1982, pp. 200-205.

L. J. Siegel, H. J. Siegel, and A. E. Feather, "Parallel processing approaches to image correlation," *IEEE Trans. Computers*, Vol. C-31, Mar. 1982, pp. 208-218.

L. J. Siegel, H. J. Siegel, and P. H. Swain, "Performance measures for evaluating algorithms for SIMD machines," _IEEE Trans. Software Engineering_, Vol. SE-8, July 1982, pp. 319-331.

M. R. Warpenburg and L. J. Siegel, "Image resampling in an SIMD environment, _IEEE Trans. Computers_, Oct. 1982, pp. 934-942.

H. J. Siegel, P. H. Swain, and B. W. Smith, "Remote sensing on PASM and CDC Flexible Processors," in _Multicomputers and Image Processing: Algorithms and Programs_, K. Preston and L. Uhr, eds. Academic Press, New York, NY, 1982, pp. 331-342.

D. L. Tuomenoksa, G. B. Adams III, H. J. Siegel, and O. R. Mitchell, "A parallel algorithm for contour extraction: advantages and architectural implications," _1983 IEEE Comp. Soc. Symp. Computer Vision and Pattern Recognition_, June 1983, pp. 336-344.

Papers Supported by AFOSR Grant No. AFOSR-78-3581

H. J. Siegel
October 1983

The following 4 research book chapters, 12 journal papers, and 37 conference papers were supported in part or full by my Air Force Office of Scientific Research Grant No. AFOSR-78-3581, which began March 1, 1978.

Research Book Chapters

[1] Howard Jay Siegel, "PASM: A Reconfigurable Multi-microcomputer System for Image Processing," in *Languages and Architectures for Image Processing,* edited by M.J.B. Duff and S. Levialdi, Academic Press, London, pp. 257-265,

[2] Leah J. Siegel, Howard Jay Siegel, and Philip H. Swain, "Parallel Algorithm Performance Measures," in *Multicomputers and Image Processing: Algorithms and Programs,* edited by K. Preston and L. Uhr, Academic Press, New York, New York, pp. 241-252, 1982.

[3] Howard Jay Siegel, Philip H. Swain, and Bradley W. Smith, "Remote Sensing on PASM and CDC Flexible Processors," in *Multicomputers and Image Processing: Algorithms and Programs,* edited by K. Preston and L. Uhr, Academic Press, New York, New York, pp. 331-342, 1982.

[4] Howard Jay Siegel, "The PASM System and Parallel Image Processing," in *Computer Architectures for Spatially Distributed Data,* edited by H. Freeman and G. G. Pieroni, Springer-Verlag, New York, NY, 1983, to appear.

Serial Journal Regular Articles

[1] Howard Jay Siegel, "Interconnection Networks for SIMD Machines," *Computer,* Vol. 12, No. 6, pp. 57-65, June 1979. (Reprinted in: (1) *Tutorial: Distributed Processor Communication Architecture,* edited by K. J. Thurber, IEEE, New York, NY, 1979, pp. 379-387, and (2) *Tutorial on Parallel Processing,* edited by R. Kuhn and D. A. Padua, IEEE Computer Society Press, New York, NY, 1981, pp. 110-119.)

[2] Howard Jay Siegel, Robert J. McMillen, and Philip T. Mueller, Jr., "A Survey of Interconnection Methods for Reconfigurable Parallel Processing Systems," *Nikkei Electronics* (Japanese publication), No. 228, pp. 49-83, December 1979 (translated into Japanese from 1979 National Computer Conference paper).

[3] Howard Jay Siegel, "A Model of SIMD Machines and a Comparison of Various Interconnection Networks," *IEEE Transactions on Computers,* Vol. C-28, No. 12, pp. 907-917, December 1979.

[4] Howard Jay Siegel, "The Theory Underlying the Partitioning of Permutation Networks," *IEEE Transactions on Computers,* Vol. C-29, No. 9, pp. 791-801,

September 1980.

[5] Howard Jay Siegel and Robert J. McMillen, "Using the Augmented Data Manipulator Network in PASM," *Computer,* Vol. 14, No. 2, pp. 25-33, February 1981.

[6] Howard Jay Siegel, Leah J. Siegel, Frederick Kemmerer, Philip T. Mueller, Jr., Harold E. Smalley, Jr., and S. Diane Smith, "PASM: A Partitionable SIMD/MIMD System for Image Processing and Pattern Recognition," *IEEE Transactions on Computers,* Vol. C-30, No. 12, pp. 934-947, December 1981.

[7] Howard Jay Siegel and Robert J. McMillen, "The Multistage Cube: A Versatile Interconnection Network," *Computer,* Vol. 14, No. 12, pp. 65-76, December 1981.

[8] Leah J. Siegel, Howard Jay Siegel, and Arthur E. Feather, "Parallel Processing Approaches to Image Correlation," *IEEE Transactions on Computers,* Vol. C-31, No. 3, pp. 208-218, March 1982.

[9] George B. Adams III and Howard Jay Siegel, "On the Number of Permutations Performable by the Augmented Data Manipulator Network," *IEEE Transactions on Computers,* Vol. C-31, No. 4, pp. 270-277, April 1982.

[10] George B. Adams III and Howard Jay Siegel, "The Extra Stage Cube: A Fault Tolerant Interconnection Network for Supersystems," *IEEE Transactions on Computers,* Vol. C-31, No. 5, pp. 443-454, May 1982.

[11] Leah J. Siegel, Howard Jay Siegel, and Philip H. Swain, "Performance Measures for Evaluating Algorithms for SIMD Machines," *IEEE Transactions on Software Engineering,* Vol. SE-8, No. 4, pp. 319-331, July 1982.

[12] Robert J. McMillen and Howard Jay Siegel, "Routing Schemes for the Augmented Data Manipulator Network in an MIMD System," *IEEE Transactions on Computers,* Vol. C-31, No. 12, pp. 1202-1214, December 1982.

Conference Proceedings

[1] S. Diane Smith and Howard Jay Siegel, "Recirculating, Pipelined, and Multistage SIMD Interconnection Networks," *Proceedings of the 1978 International Conference on Parallel Processing* (IEEE Catalog No. 78CH1321-9), pp. 206-214, Bellaire, Michigan, August 1978.

[2] Howard Jay Siegel, Philip T. Mueller, Jr., and Harold E. Smalley, Jr., "Control of a Partitionable Multimicroprocessor System," *Proceedings of the 1978 International Conference on Parallel Processing* (IEEE Catalog No. 78CH1321-9), pp. 9-17, Bellaire, Michigan, August 1978.

[3] Howard Jay Siegel and Philip T. Mueller, Jr., "The Organization and Language Design of Microprocessors for an SIMD/MIMD System," *Proceedings of the Second Rocky Mountain Symposium on Microcomputers: Systems, Software, Architecture* (IEEE Catalog No. 78CH1387-0), pp. 311-340, Pingree Park, Colorado, August 1978.

[4] Howard Jay Siegel, "Partitionable SIMD Computer System Interconnection Network Universality," *Proceedings of the Sixteenth Annual Allerton Conference on Communication, Control, and Computing,* University of Illinois-Urbana, pp. 586-595, Monticello, Illinois, October 1978.

[5] S. Diane Smith and Howard Jay Siegel, "An Emulator Network for SIMD Machine Interconnection Networks," *Proceedings of the Sixth Annual International Symposium on Computer Architecture* (IEEE Catalog No. 79CH1394-6),

pp. 232-241, Philadelphia, Pennsylvania, April 1979.

[6] Howard Jay Siegel, Robert J. McMillen, and Philip T. Mueller, Jr., "A Survey of Interconnection Methods for Reconfigurable Parallel Processing Systems," *AFIPS Conference Proceedings Volume 48: 1979 National Computer Conference*, pp. 529-542, New York City, New York, June 1979. (Translated into Japanese and reprinted in *Nikkei Electronics*, No. 228, pp. 49-83, December 1979.)

[7] Howard Jay Siegel, Leah J. Siegel, Robert J. McMillen, Philip T. Mueller, Jr., and S. Diane Smith, "An SIMD/MIMD Multimicroprocessor System for Image Processing and Pattern Recognition," *Proceedings of the 1979 IEEE Computer Society Conference on Pattern Recognition and Image Processing (PRIP 79)* (IEEE Catalog No. 79CH1428-2), pp. 214-224, Chicago, Illinois, August 1979.

[8] Howard Jay Siegel, "Partitioning Permutation Networks: The Underlying Theory," *Proceedings of the 1979 International Conference on Parallel Processing* (IEEE Catalog No. 79CH1433-2), pp. 175-184, Bellaire, Michigan, August 1979.

[9] Howard Jay Siegel, Frederick Kemmerer, and Mark Washburn, "Parallel Memory System for a Partitionable SIMD/MIMD Machine," *Proceedings of the 1979 International Conference on Parallel Processing* (IEEE Catalog No. 79CH1433-2), pp. 212-221, Bellaire, Michigan, August 1979.

[10] Howard Jay Siegel and S. Diane Smith, "An Interconnection Network for Multimicroprocessor Emulator Systems," *Proceedings of the First International Conference on Distributed Computing Systems* (IEEE Catalog No. 79CH1445-6), pp. 772-782, Huntsville, Alabama, October 1979.

[11] Robert J. McMillen and Howard Jay Siegel, "MIMD Machine Communications Using the Augmented Data Manipulator Network," *Proceedings of the Seventh Annual International Symposium on Computer Architecture* (IEEE Catalog No. 80CH1494-4), pp. 51-58, La Baule, France, May 1980.

[12] S. Diane Smith, Howard Jay Siegel, Robert J. McMillen, and George B. Adams III, "Use of the Augmented Data Manipulator Multistage Network for SIMD Machines," *Proceedings of the 1980 International Conference on Parallel Processing* (IEEE Catalog No. 80CH1569-3), pp. 75-78, Harbor Springs, Michigan, August 1980.

[13] Robert J. McMillen, George B. Adams III, and Howard Jay Siegel, "Permuting with the Augmented Data Manipulator Network," *Proceedings of the Eighteenth Annual Allerton Conference on Communication, Control, and Computing*, University of Illinois-Urbana, pp. 544-553, Monticello, Illinois, October 1980.

[14] Philip T. Mueller, Jr., Leah J. Siegel, and Howard Jay Siegel, "A Parallel Language for Image and Speech Processing," *Proceedings of the IEEE Computer Society's Fourth International Conference on Computer Software and Applications (COMPSAC '80)* (IEEE Catalog No. 80CH1607-1), pp. 476-483, Chicago, Illinois, October 1980.

[15] Robert J. McMillen and Howard Jay Siegel, "The Hybrid Cube Network," *Proceedings of the Distributed Data Acquisition, Computing. and Control Symposium* (IEEE Catalog No. 80CH 1571-9), pp. 11-22, Miami Beach, Florida, December 1980.

[16] Howard Jay Siegel and Robert J. McMillen, "The Use of the Augmented Data Manipulator Network in PASM," *Proceedings of the Fourteenth Annual Hawaii International Conference on System Sciences*, pp. 228-237, Honolulu, Hawaii,

January 1981.

[17] Robert J. McMillen and Howard Jay Siegel, "Dynamic Rerouting Tag Schemes for the Augmented Data Manipulator Network," *Proceedings of the Eighth Annual International Symposium on Computer Architecture* (IEEE Catalog No. 81CH1593-3), pp. 505-516, Minneapolis, Minnesota, May 1981.

[18] Robert J. McMillen, George B. Adams III, and Howard Jay Siegel, "Performance and Implementation of 4x4 Switching Nodes in an Interconnection Network for PASM," *Proceedings of the 1981 International Conference on Parallel Processing* (IEEE Catalog Number 81CH1634-5), pp. 229-233, Bellaire, Michigan, August 1981.

[19] Leah J. Siegel, Howard Jay Siegel, and Arthur E. Feather, "Parallel Image Correlation," *Proceedings of the 1981 International Conference on Parallel Processing* (IEEE Catalog Number 81CH1634-5), pp. 190-198, Bellaire, Michigan, August 1981.

[20] Howard Jay Siegel and Philip H. Swain, "Contextural Classification on PASM," *Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing (PRIP 81)* (IEEE Catalog Number 81CH1595-8), pp. 320-325, Dallas, Texas, August 1981.

[21] Leah J. Siegel, Edward J. Delp, Trevor N. Mudge, and Howard Jay Siegel, "Block Truncation Coding on PASM," *Proceedings of the Nineteenth Annual Allerton Conference on Communication, Control, and Computing,* University of Illinois-Urbana, pp. 891-900, Monticello, Illinois, October 1981.

[22] David L. Tuomenoksa and Howard Jay Siegel, "Application of Two-Dimensional Bin Packing Algorithms for Task Scheduling in the PASM Multimicrocomputer System," *Proceedings of the Nineteenth Annual Allerton Conference on Communication, Control, and Computing,* University of Illinois-Urbana, pg. 542, Monticello, Illinois, October 1981.

[23] James T. Kuehn and Howard Jay Siegel, "Simulation Studies of PASM in SIMD Mode," *Proceedings of the 1981 IEEE Computer Society Workshop on Computer Architecture for Pattern Analysis and Image Database Management* (IEEE Catalog No. 81CH1697-2), pp. 43-50, Hot Springs, Virginia, November 1981.

[24] George B. Adams III and Howard Jay Siegel, "A Multistage Network with an Additional Stage for Fault Tolerance," *Proceedings of the Fifteenth Annual Hawaii International Conference on System Sciences,* pp. 333-342, Honolulu, Hawaii, January 1982.

[25] Robert J. McMillen and Howard Jay Siegel, "Performance and Fault Tolerance Improvements in the Inverse Augmented Data Manipulator Network," *Proceedings of the Ninth Annual International Symposium on Computer Architecture* (IEEE Catalog Number 82CH1754-1), pp. 63-72, Austin, Texas, April 1982.

[26] George B. Adams III and Howard Jay Siegel, "Properties of the Extra Stage Cube Under Multiple Faults," *Proceedings of the Fourteenth Southeastern Symposium on System Theory* (IEEE Catalog Number 82CH1752-5), Blacksburg, Virginia, pp. 3-6, April 1982.

[27] Edward J. Delp, Trevor N. Mudge, Leah J. Siegel, and Howard Jay Siegel, "Parallel Processing for Computer Vision," *Society of Photo-Optical Instrumentation Engineers Proceedings Vol. 336: Robot Vision,* pp. 161-167, Arling-

ton, Virginia, May 1982.

[28] Trevor N. Mudge, Edward J. Delp, Leah J. Siegel, and Howard Jay Siegel, "Image Coding Using the Multimicroprocessor System PASM," *Proceedings of the 1982 IEEE Computer Society Conference on Pattern Recognition and Image Processing* (IEEE Catalog No. 82CH1761-6), pp. 200-207, Las Vegas, Nevada, June 1982.

[29] David Lee Tuomenoksa and Howard Jay Siegel, "Analysis of the PASM Control System Memory Hierarchy," *Proceedings of the 1982 International Conference on Parallel Processing* (IEEE Catalog No. 82CH1794-7), pp. 363-370, Bellaire, Michigan, August 1982.

[30] James T. Kuehn, Howard Jay Siegel, and Peter D. Hallenbeck, "Design and Simulation of an MC68000-Based Multimicroprocessor System," *Proceedings of the 1982 International Conference on Parallel Processing* (IEEE Catalog No. 82CH1794-7), pp. 353-362, Bellaire, Michigan, August 1982.

[31] Robert J. McMillen and Howard Jay Siegel, "A Comparison of Cube Type and Data Manipulator Type Networks," *Proceedings of the 3rd International Conference on Distributed Computing Systems* (IEEE Catalog No. 82CH1802-8), pp. 614-621, Hollywood, Florida, October 1982.

[32] David Lee Tuomenoksa and Howard Jay Siegel, "Analysis of Multiple-Queue Task Scheduling Algorithms for Multiple-SIMD Machines," *Proceedings of the 3rd International Conference on Distributed Computing Systems* (IEEE Catalog No. 82CH1802-8), pp. 114-121, Hollywood, Florida, October 1982.

[33] Howard Jay Siegel, O. Robert Mitchell, and Leah J. Siegel, "PASM: A Large-Scale System for Studying Parallel Image Processing," *1982 Government Microcircuits Applications Conference Digest of Papers*, pp. 186-189, Orlando, Florida, November 1982.

[34] David Lee Tuomenoksa, George B. Adams, III, Howard Jay Siegel, and O. Robert Mitchell, "A Parallel Algorithm for Contour Extraction: Advantages and Architectural Implications," *Proceedings of the 1983 IEEE Computer Society Symposium on Computer Vision and Pattern Recognition (CVPR)* (IEEE Catalog No. 83CH1891-1), pp. 336-374, Arlington, Virginia, June 1983.

[35] David Lee Tuomenoksa and Howard Jay Siegel, "Preloading Schemes for the PASM Parallel Memory System," *Proceedings of the 1983 International Conference on Parallel Processing* (IEEE Catalog No. 83CH1922-4), pp. 407-415, Bellaire, Michigan, August 1983.

[36] James T. Kuehn, Howard Jay Siegel, and Martin Grosz, "A Distributed Memory Management System for PASM," *Proceedings of the 1983 IEEE Computer Society Workshop on Computer Architecture for Pattern Analysis and Image Database Management*, Pasadena, California, October 1983, to appear.

[37] David L. Tuomenoksa and Howard Jay Siegel, "A Distributed Operating System for PASM," *Proceedings of the Seventeenth Hawaii International Conference on System Sciences*, Honolulu, Hawaii, January 1984, to appear.

EN

FILM

2 -8

DTI