HIGH DENSITY PARALLEL PROCESSING

I. The Processor Array and Macro Controller

H. T. Nguyen, R. Raghavan, C. H. Ting, H. S. Truong

Lockheed Palo Alto Research Laboratories, Palo Alto, CA

Summary

A GAPP processor array of 11520 processors and its associated controller were built and tested. It allows the processor array to be programmed conveniently using high level languages without sacrificing speed or code efficiency. The system is fully functional. The hardware structure and special features of this system are presented in this report.

1. The Parallel Processor System.

As part of the process of evaluating parallel processor algorithms with emphasis on image processing we have developed a complete parallel array processor system. This is a necessary tool because large programs require unacceptable long time to execute on a software simulator, and because algorithm optimization ultimately requires testing with real time data. Included in the system are a 96 by 120 array of GAPP processors(Geometric Arithmetic Parallel Processor, NCR 45CG72) and an MIMD (Multiple Instruction Multiple Datapaths) controller optimized for program compression and fast program flow control.

Rather than inventing a new operating environment, the system was designed to operate as an external coprocessor to an IBM AT personal computer as a host. Paths are provided from the host to the parallel processor system for program and data loading, run-time operation, and status monitoring. In addition, high speed 12 bit parallel input and output ports are provided which are capable of 10 megawords per second syncrhronous data transfers, and slower asynchronous transfers.

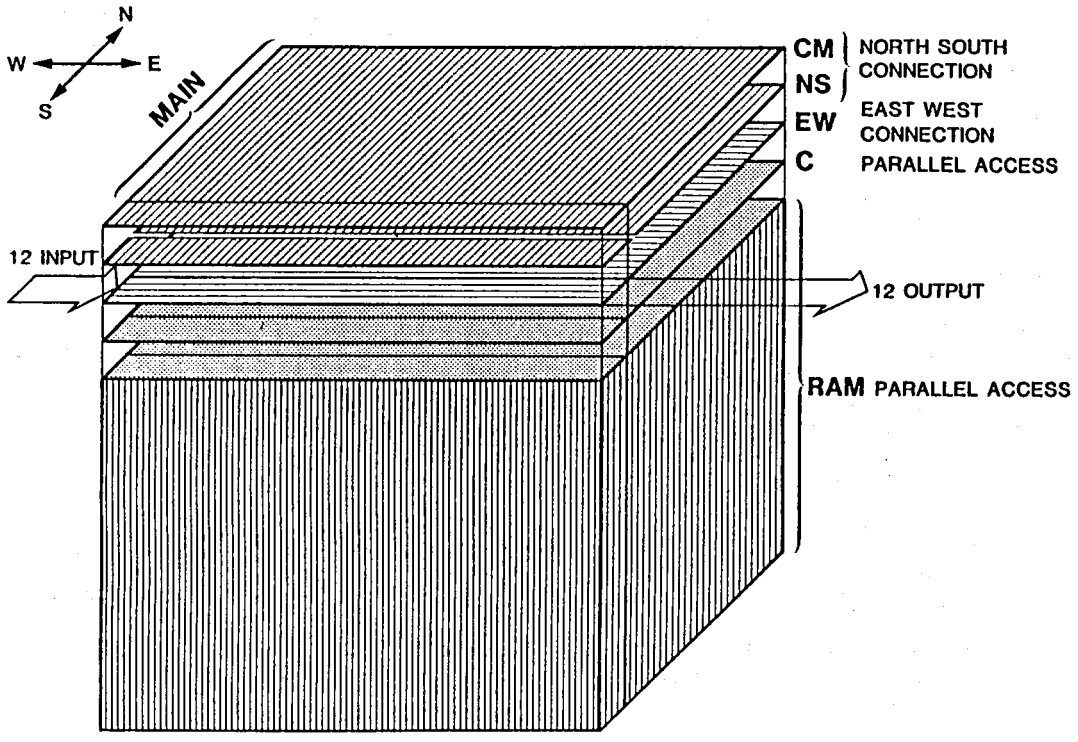
Processing within the parallel processor system is completely self-contained so that once started by the host, program execution can proceed independently. Our present system loads data via a DMA channel in the host computer and the results can be unloaded similarly to host or to a real time video display unit. A software console program was developed for use in the host computer to control the parallel processor system interactively.

2. Processor Array

The SIMD (Single Instruction Multiple Datapaths) data processing section consists of an array of GAPP devices and is constructed from four circuit board assemblies, each of which has a 60 by 48 array of single bit GAPP processors. The boards, which were specially designed for this application, contain extensive signal buffering to allow array expansion in all four directions by using multiple boards. This versatility allows altering the array aspect ratio for experimentation with various classes of problems. Array expansion with these components is feasible up to approximately 256 by 256 cells at which point it is worthwhile to design a unique board package for each case so as to optimize density and area efficiency.

The processors are organized as two arrays as show in Figure 1: one of 12 by 96 processors for input/output corner-turning, and a main array of 108 by 96 processors. The two arrays may optionally execute from independent instruction and address streams. However, in the present system only one stream is used. In the main array, the EW and NS register planes are connected in a cylindrical surface topology in both the east-west and north-south directions, although spiral and other interconnections are jumper selectable. The corner-turning array is cylindrical in the north-south direction for the NS register plane, and uses the east edges for input and the west edge for output. The two arrays are connected, also in a cylindrical manner, by the CM register plane in the GAPP devices.

Figure 1. GAPP processor array.

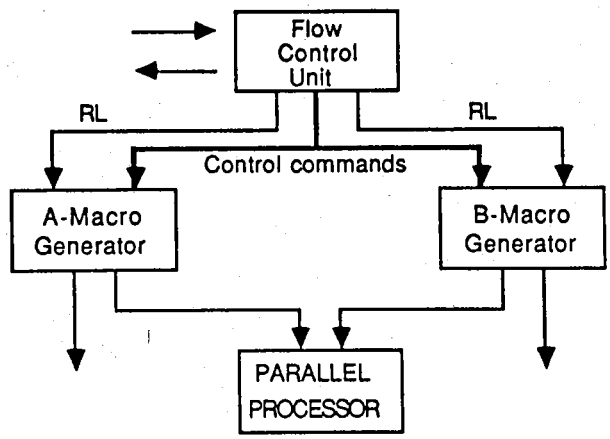


3. The Distributed Macro Controller (DMC)

The controller, dubbed DMC, addresses the critical issue in parallel processing computers like the GAPP with high processor density and limited memory and instruction sets. The controller allows ready implementation of adaptive programming decisions made by the host; that is, without loss of machine cycles. The top level architectural innovation is that the controller is a MIMD machine that processes three different instructions streams simultaneously as show in Figure 2. A Flow Control Unit feeds several (here two) Macro Generator Units. The instruction streams from the Macro Generator Units are combined to feed the control lines of the GAPP array. Each of these units will be a single chip in VLSI. All Macro Generator Units are identical.

Both the Flow Control and Macro Generator Units use externally writable control stores to store instruction streams. The Flow Control Unit supervises program flow within DMC while the Macro Generator Units produce output instructions for the GAPP array. The MIMD architecture is hierarchical; i.e., the Flow Control Unit directs the production of the programs from the Macro Generator Units. The final output stream consists of two 15 bit words, combined to form a single 20 bit instruction and address stream for the GAPP. The controller offers a very high degree of program compression. Existing sequencers have wide microcode words but little program optimization or compression.

The Flow Control Unit allows eight levels of nested subroutines and eight levels of nested loops. While loops increment, the loop counts of interior loops can be changed. Subroutine calls and returns are performed in 3 clock cycles. With the provision that subroutines are at least three instructions long, this allows penalty-free macroprogramming. External inputs may be tested for conditional operatings (branching, looping, calling, and returning). The Flow Control Unit uses a 32 bit wide instruction format. It is designed to be a single chip unlike existing sequencers, although its primary function in the present controller is to direct the internal flow (within the DMC) of the program.



RL: Reinterpretation Logic

Figure 2. Functional units within the DMC.

The Macro Generator Units, which are physically identical and each designed to be a single chip, have several novel features:

(1) Callable macro and address routines
(2) Automatic memory management
(3) Static and dynamic reinterpretation logic
(4) A rich set of stack operation.

Feature (1) is for program compression. Pre-loaded instruction streams can be called by specifying a pointer and length. Typically, these are not GAPP instructions but instructions which cause the Macro Generator Units to produce GAPP code indirectly.

Memory management calculates physical addresses given logical ones. Thus all of the memory addressing is indirect and penalty free. A linked list of memory segments with "occupied" and "free" areas is maintained. This handles allocation of memory and makes the task of the GAPP programmer much easier. Also, if these functions were to be performed in software, the processing system would not be able to operate at full speed.

Reinterpretation is a method of program compression useful from both an op-code and memory point of view. There are both dynamic and static reinterpretations available. Reinterpretation involves performing an exclusive-OR operation on the output with a mask pattern. A number of patterns may be stored. Dynamic reinterpretation allows an external constant to be loaded in where the bits of the constant can be used to modify the output with one of several masks. Static reinterpretation is only selectable at the macro-instruction level. Thus, the if-then-else construction becomes available to parallel processors without penalty.

The usefulness of reinterpretation is that applications natural to geometric SIMD machines tend to be highly patterned. Addition, subtraction and template matching to either a zero or one differ only in the selection of CARRY and BORROW, loops proceed by alternately selecting one stack or another, etc. A study of the class of transformations natural to GAPP-like machines reveals the frequent occurrence of such patterns allowing switching between instructions with the use of reinterpretation bits. Reinterpretation of address bits allows symmetrical operations within address space.

The controller also provides a rich set of stack operators, operating simultaneously on two stacks holding address pointers to the GAPP memory. Stacks offer a way of changing the instruction sequence to the GAPP in nonconsecutive or nonlinear ways. Two stacks with two top elements cached in the address Macro Generator Unit give the programmer convenient access to four different memory areas to implement complex arithmetic and logic operations.

4. Conclusions

This parallel processor system is currently fully functional with the GAPP processor array. The system was designed to run at the maximum clock rate of 10 Mhz. Currently it is running at 2.5 Mhz. The computational throughput is thus 28.8 Giga instructions per second. We are developing software tools and programs to evaluate its performance for many different classes of problems, such as image processing and understanding, real time signal processing and analysis, translation invariant and non-invariant problems, and the general studies on the using and programming of parallel processing systems. Some of the results will be show in a video tape, demostrating the real time image processing capability of this system.

The controller is rich in ways that optimize the programming of GAPP-like arrays. The detailed architecture will be presented in a patent application. We believe that the study of its concepts will allow better understanding of the maximum efficiency obtainable from geometric SIMD arrays and lead to distinct developments in this branch of computer science.

5. Acknowledgements

Dr. Wlodzimierz Holsztynski, the inventor of the GAPP device, provided the architectural design of the Distributed Macro Controller. Integrated Test Systems of Santa Barbara, CA constructed both the GAPP processor boards and the Distributed Macro Controller system.