Fermi National Accelerator Laboratory

FERMILAB-Conf-84/63
2380.000

THE FERMILAB ACP MULTI-MICROPROCESSOR PROJECT*

I. Gaines, H. Areti, J. Biel, S. Bracker,
G. Case, M. Fischler, D. Husby, and T. Nash

August 1984

*Presented at the Symposium on Recent Developments in Computing, Processor, and Software Research for High-Energy Physics, Guanajuato, Mexico, May 8-11, 1984

The Fermilab ACP Multi-Microprocessor Project

I. Gaines, H. Areti, J. Biel, S. Bracker, G. Case,
M. Fischler, D. Husby, T. Nash

Advanced Computer Program
Fermi National Accelerator Laboratory
Batavia, Illinois 60510

ABSTRACT

We report on the status of the Fermilab Advanced Computer Program's project to provide more cost-effective computing engines for the high energy physics community. The project will exploit the cheap, but powerful, commercial microprocessors now available by constructing modular multi-microprocessor systems. A working test bed system as well as plans for the next stages of the project are described.

Introduction

High energy physics experiments have become more and more complex and are accumulating ever increasing amounts of data. The need for computing to analyze these experiments has expanded enormously. Elsewhere in high energy physics computing problems, such as beam-orbit simulations for the design of the SSC and lattice-guage theory calculations, are also expected to require large amounts of computing time. We can no longer afford enough conventional computers for the overall high energy physics workload. Many experiments have already had their ability to do physics compromised by limitations in the amount of off-line computing power made available to them. With the turn-on of a number of even more complex colliding beam detectors in the immediate future, the problem has become so acute that it has spawned several high level review committees.

In response to this problem, Fermilab has established the Advanced Computer Program (ACP) [1] with the primary mission of developing new approaches to computing that will represent more cost-effective alternatives to conventional mainframes for the compute-bound problems of high energy physics. The ACP's first project is the development of a flexible and modular approach to multiprocessing based on 32 bit microprocessors of near VAX class power. We describe this project, its goals, plans, and status, in the following.

Design Goals and Concepts

One method of providing more cost-effective computing is to design dedicated special purpose processors for particular problems. In the high energy physics community such devices are in common use as trigger processors. [2] Such devices have almost no limit to the increase in cost-effectiveness that can be provided, but suffer from the disadvantage of being relatively inflexible and difficult to program. Changing to a different algorithm requires a large amount of work by system experts.

On the other hand, commercial computer manufacturers and university computer scientists usually focus on designs of fully general parallel processing systems, where large numbers of processors can all be brought to bear on an arbitrarily general problem. Such fully general systems must solve the difficult problems of shared memory, interconnection networks, and synchronization mechanisms. The complexity inherent in the goal of generality implies a long delay in bringing the designs to practical fruition. Furthermore, much of the cost of such systems goes into pieces other than the processing elements themselves, reducing the potential cost-effectiveness.

The ACP project is neither fully general nor dedicated special purpose. Rather, it is attempting to exploit the characteristics of the relatively well understood high energy physics computing problems to design a simple and straightforward architecture that gives near maximal cost-effectiveness for these problems while maintaining the flexibility and programability of general purpose computers. In particular, the most important feature of the high energy experiment computing problems is their event oriented nature. A typical experiment may have tens or hundreds of millions of events, each of which is an essentially independent analysis problem. The natural and trivial parallelism inherent in the problem leads to a multiprocessor solution with no global memory and simple interconnections, but where each processor has sufficient local memory to process a complete single event.

The ACP project will exploit additional characteristics of the problem to yield improved cost-effectiveness. These include the existence of compute-bound kernels (inner loops in the programs which use very large fractions of the overall CPU time), structured blocking in the programs with minimal communication between the blocks, and very long (weeks to months) run times for the same program on different data tapes. This makes it sensible to design special purpose "hardware subroutine" coprocessors for efficient execution of the inner loops of particular types of problems. It is also appropriate to allow for reconfiguring the connection topology and the distribution of memory and special coprocessors for the needs of a particular program with a long production run.

The critical goal of very high cost-effectiveness for the ACP system, therefore, is met by the following features of the design: an extremely simple architecture; small, mass-produced VLSI (and thus cheap) CPUs; and (eventually) from high-speed special purpose hardware attached to the CPUs for particular problems. Another important design goal is modularity, which allows the system to be optimally reconfigured for a given problem and allows the use of newer and faster CPUs and other components without redesigning the entire system. For this, it is important to construct the system out of commercially available VLSI and board level components whenever possible. This reduces initial design effort, can reduce costs and will make it easier to make copies of the system with minimal expert assistance. A third important goal is user friendliness, which is realized by supporting FORTRAN-77 on processing CPUs and making available program development and debugging tools on a convenient host machine.

The ACP system can be summarized in a long-winded phrase, as a flexible, loosely-coupled multi-microprocessor system, with optional customized special purpose hardware subroutine coprocessors. It is broadly applicable to a large class of compute-bound problems which share the important characteristic of being "event-oriented," that is, having a natural simple parallelism inherent in

the problem. These include a number ouside of high energy physics such as process simulation, robotics, animation, and finite element analysis.

System Overview and Phasing [3]

The core of the ACP system is the individual processing node (shown with some optional additions in Figure 1). The node always consists of a processor which supports user software written in a high-level language (FORTRAN-77) and sufficient memory to contain an entire event (at least 1 Mbyte). Optionally, as required for particular problems, the node may also contain additional memory up to 16 Mbytes, floating point hardware coprocessors, special purpose coprocessors optimized for the compute-bound kernel running on that node, and nearest neighbor node communication interfaces for problems requiring fast grid-like internode communication.

Each node lives within a dual bus structure. It is a slave on a global bus over which programs and data are downloaded to all the nodes. The node's CPU accesses its own local memory as a master over a private local bus. Thus, each node can address its own memory simultaneously without any contention on the global bus. High-speed hardware coprocessors may even require a third super-fast bus to process data in memory with a much faster cycle time than that of the local bus.

The software within such a node is simple because the node is a slave on the global bus. The node waits for events to be delivered to it and processes them on command. The primitive "operating system" which runs on the individual nodes must only support the FORTRAN run time environment (but not I/O), trap exceptions, and handle communication with a host CPU through dedicated memory locations. This node software system jumps to the user code when a flag is set indicating the presence of an event. It sets a second flag indicating completion when the user code returns. Further details on ACP work on support software are found in the companion paper, "User Software for Event-Oriented Processing" by M. Fischler et al. [4].

Arrays of such nodes can be configured in a variety of topologies, depending on the problem at hand. These range from the most simple (Figure 2) where a collection of identical processors are lined up each to receive individual events, to the more complicated arrangement of multiple ranks of processors shown in Figure 3. Other arrangements, suitable for accelerator beam-orbit simulations, are discussed in Reference 5.

We require a CPU node to have the processing power for reconstruction codes of at least 0.5 VAX 11/780 (or else too many nodes are required), and to run high level language programs (specifically, FORTRAN-77). It should use cheap memory technology (high production MOS dynamic RAMs) so that comfortable amounts of memory can be made available in each node. Upward compatability to higher performance parts without major system redesign is also required as is easy coprocessor interface. All of these considerations point clearly to the use of commercial microprocessors for the CPU nodes, provided they can meet the performance goals.

Fortunately, at least six different vendors (AT & T, DEC, Intel, Motorola, National Semiconductor, and Zilog) have announced 32 bit microprocessors with expected performance well above the ACP goals. Three vendors (AT & T, Motorola, and National) already have working 32 bit chips. The ACP group has benchmarked

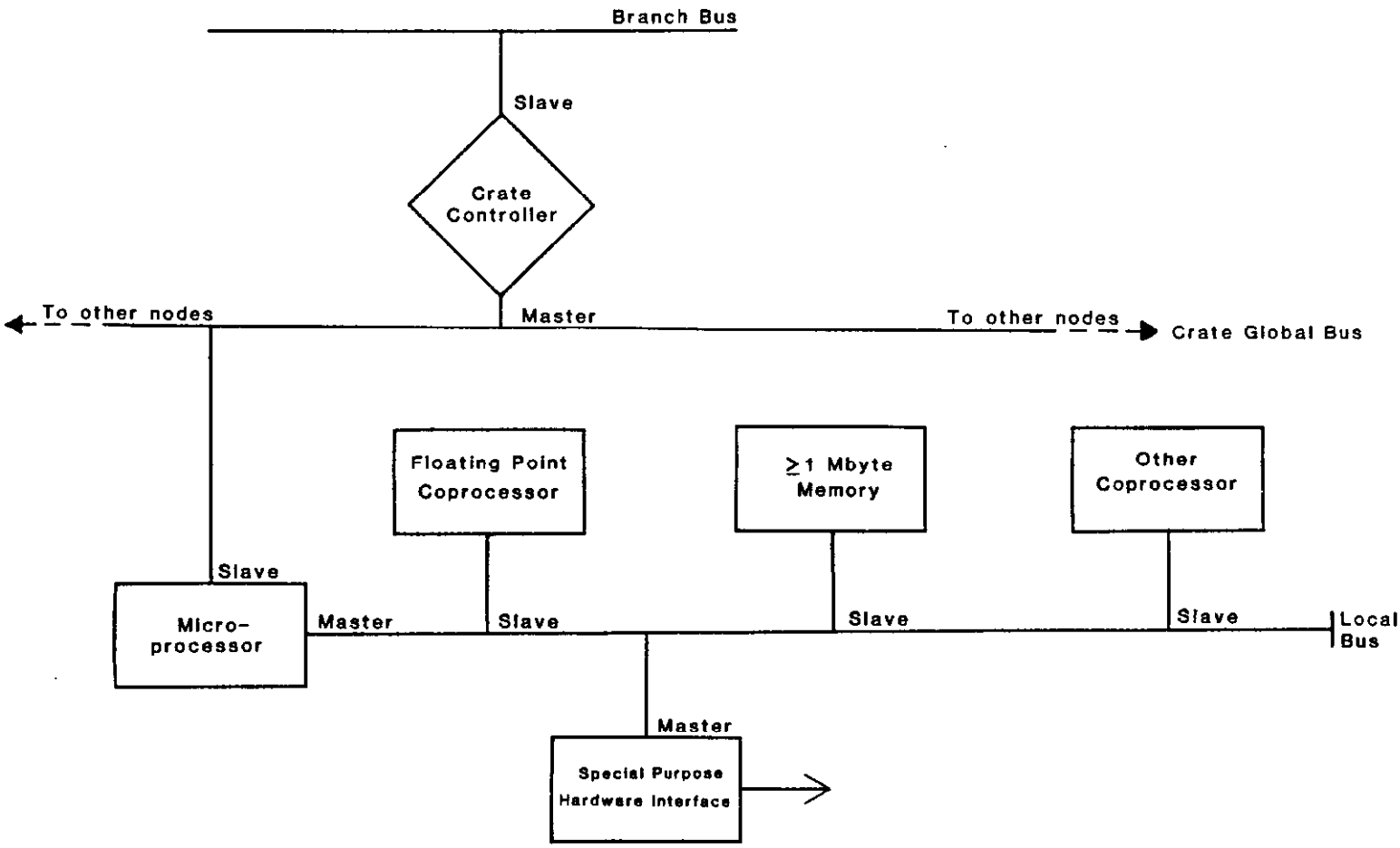

Figure 1. A single processing node.

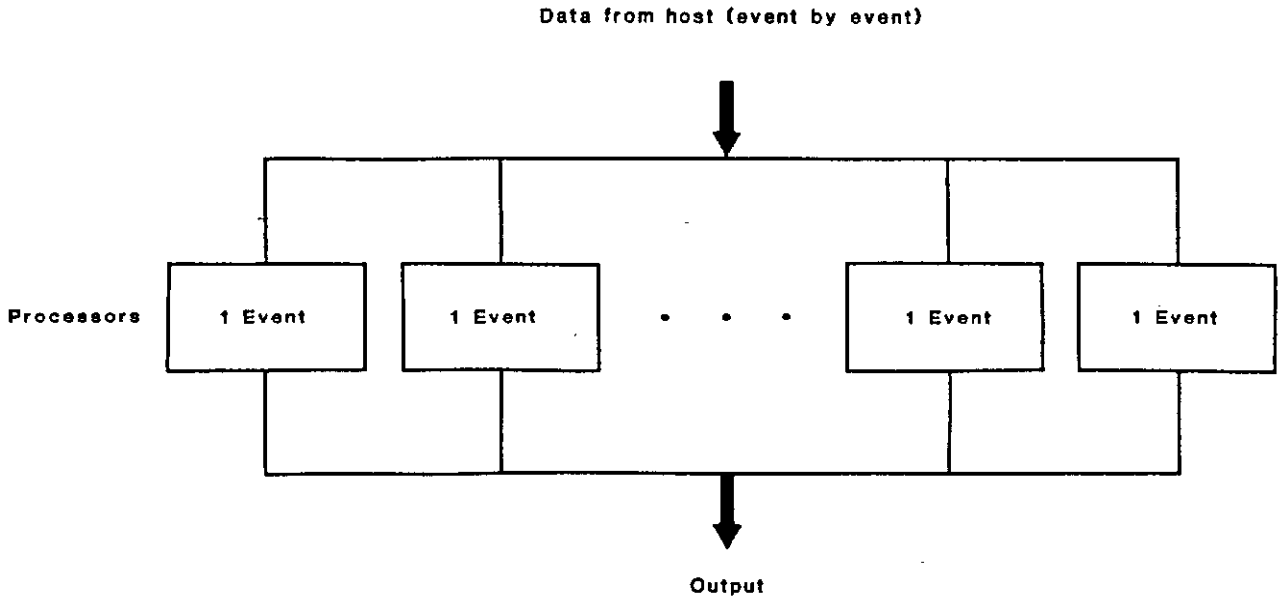

Figure 2. Single rank multiprocessor concept.

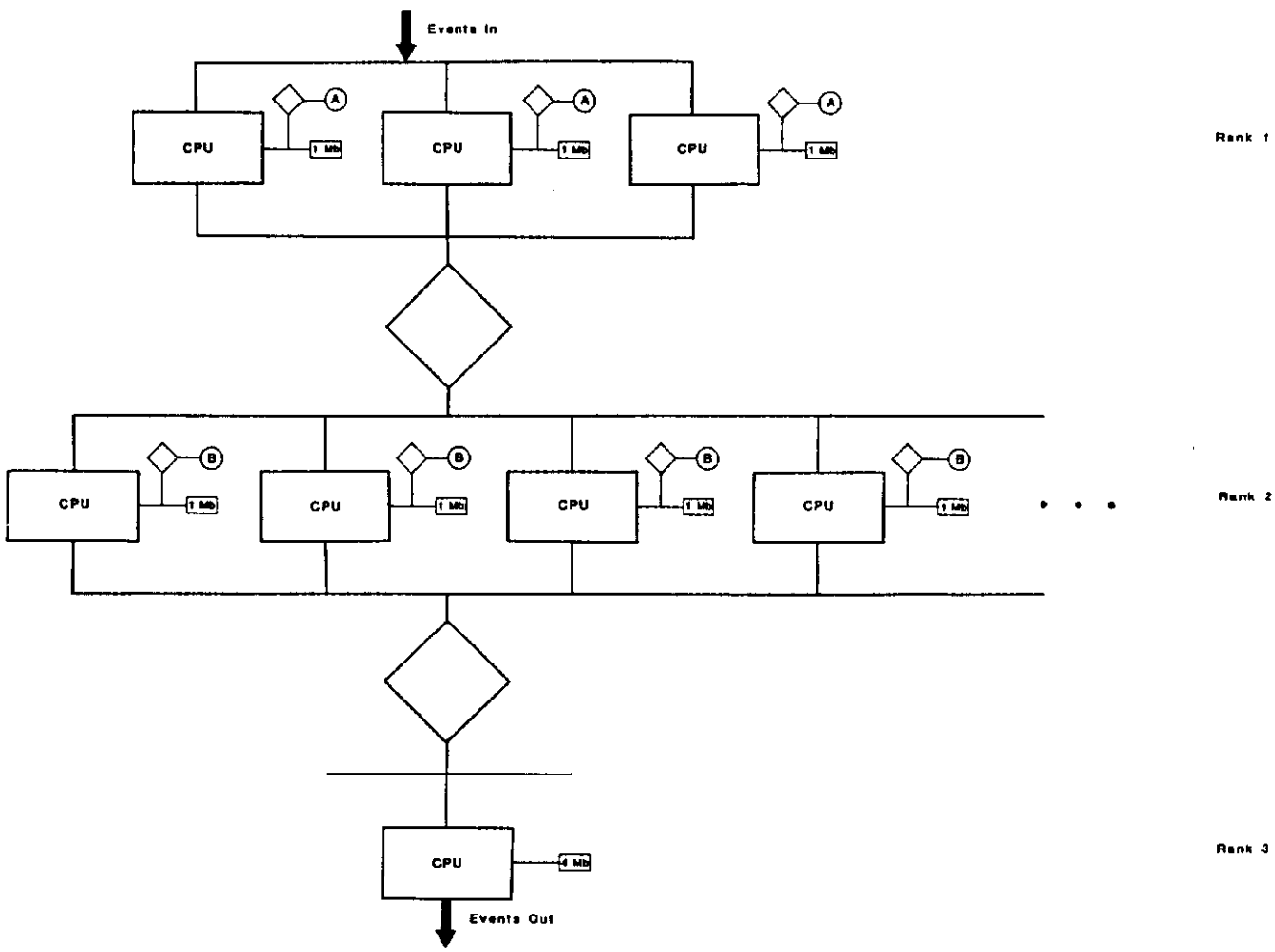

Figure 3. Multiple ranks of processors; different coprocessors and memory capacity in nodes of different ranks.

a full-scale physics track reconstruction program written in FORTRAN on existing 16 bit processors. Performance was measured relative to a VAX 11/780 as 0.1 for the Motorola 68000, and 0.12 for an Intel 80286. Taking into account the performance improvements available in the 32 bit versions of these chips, it is clear that the 32 bit commercial processors to be widely available in early 1985 will easily meet the ACP performance goals. As of this writing, we are in the process of benchmarking the new 32 bit Motorola 68020.

The project is proceeding in three phases, each of which is described more fully in subsequent sections. Phase I, now all but complete, consists of the development of support and error handling software on a test bed system using 16 bit processors and a 16 bit bus. Phase II, scheduled to be complete in summer 1985, is the first full-scale production system, consisting of at least 128 full-performance nodes. Phase III includes the development of special purpose hardware coprocessors, more complex node interconnection and host fuction schemes, and higher performance nodes.

System Components

The components of the ACP system in each of its phases consist of the following items:

1. Crate/bus - The crate must support at least two busses, a crate wide global bus and a separate segmented local bus for each CPU node in the crate. After Phase I, the global bus should support transfers at a rate of 20 Mbyte/sec with an address space sufficient for 16 Mbytes of memory for each node in the crate. The local bus should support memory accesses at a speed sufficient for the processors which will run with no wait states, and should have 16 Mbytes of address space. Only the crate controller needs to be a master on the global bus, while the individual node CPUs are each masters on their own local busses. The local bus should be reconfigurable to allow for different numbers of cards in each node at different times. Optional desirable features are a serial bus for low-speed or diagnostic transfers, and provision for a high speed coprocessor bus. Both MULTIBUS II from Intel and VME/VMX from Motorola are commercial busses that meet these requirements.

2. CPU board - The CPU board should be a commercial 32 bit microprocessor that is a master on its own local bus and can be controlled from the global bus. It must run FORTRAN-77 programs. The initial Phase II system will contain at least two different types of CPUs. It is expected that commercially produced boards will be available at competitive prices. The ACP is a "beta site" for a 68020 board under development by Motorola's Microsystem Division.

3. Memory board - The memory board needs to be dual-ported on the global and local busses, although the arbitration between the ports can be very simple (the global bus can be given absolute priority). It is expected that commercially produced boards will become available.

4. Crate controller - Used as the only master on the crate global bus, the crate controller must be able to do full-speed (about 20 Mbytes/sec) reads and writes to anywhere in the crate memory space. It is a slave to the host on a bus linking crates.

5. Host interface - This must provide data and control paths to allow the host CPU to download programs and event data to crates full of a total of up to 255 nodes. After Phase I, this system must link the host to up to 64 crate controllers. It may include the intelligence to find nodes available for new events and detect nodes with completed events.

6. Development host - A minicomputer supporting multiple users must be provided as a development host. Small numbers of nodes of each variety will be attached to this computer to allow the user to develop and debug programs for use in a multiprocessor environment. The system, most likely a VAX 11/780 running VMS, includes file editors, compilers, symbolic debuggers, etc. for both host and node user software.

7. Production host - Linked to the development host via a network (DECNET), the production host is a single user system supporting running programs on the multi-node system. It provides the user the functions of event input/output and control of the nodes in a transparent manner. The host portion of user programs, as well as system control functions, run in the production host. It is often referred to as the "roots" of the tree-like ACP multiprocessor system (see following discussion).

8. System software - Software components include: development tools (compilers and debuggers); user support subroutines to allow programs to be split into a host piece (which does event I/O and printout) and a node piece (which executes the CPU intensive portion of a user's code simultaneously on many nodes); diagnostic and verification tools; and simulators of the overall system. The system software runs on the development host and various components of the production host as well as on the nodes. This is more fully described in References 4 and 6.

Test Bed System

The Phase I test bed system, now in operation, was built to develop and test the user support multiprocessor software described in References 4 and 6. Since high performance was not required, it consists of low-speed 16 bit hardware. It includes a full software prototype with node "operating systems", user support subroutines, and command procedures for compiling and debugging. Error handling and verification capabilities are presently being developed.

The test bed hardware contains 6 CPU nodes: 5 Motorola 68000s and one Intel 8086 with an 8087 floating point coprocessor. The 8086 has 256 Kbytes of onboard RAM, while each 68000 has a 512 Kbyte memory on a separate card. The system is implemented in a MULTIBUS I crate, with MULTIBUS being used for the global bus. A commercial SAM bus manufactured by SGS Corporation (Milan and Phoenix) is used as the local bus for the 68000s. The 8086 has no local bus since all memory is on-board. The 68000 boards were designed and built by the ACP group, while the 8086 board and the memory boards (dual ported MULTIBUS and SAM bus) are commercial products, as is the crate. All five memory boards can be put on the local bus of a single 68000 to test programs requiring up to 2.5 Mbytes of memory. A VAX 11/780 is being used as the host for the test bed system, with a DR11W UNIBUS DMA interface connecting the host to the crate. An ACP built board interfaces the DR11W to the MULTIBUS and acts as the crate controller.

Both types of processors are supported with FORTRAN-77 compilers and run-time libraries. The 8086 compiler, from Intel, is a cross compiler which runs on the VAX. The 68000 compiler, from Absoft Inc. (Royal Oak, Michigan), is a native mode compiler which runs on one 68000 node. The 68000 software also includes a powerful interactive symbolic debugger which can be used to debug programs running on the nodes.

ACP software on the test bed system is the full complement of routines and utilities described in References 4 and 6. This includes the operating systems on the individual nodes, user subroutines to allow the user to split his program into a host and a node piece, automatic routines to download the user's code into the nodes and handle all host-node communication, command procedures on the VAX to compile and link the users programs for execution on the nodes, and VAX routines to support the 68000 compiler and the run-time system. Several different large high energy physics FORTRAN programs have run successfully on the test bed system. Test users are finding the support software convenient to use. A major reconstruction package was successfully brought up by two physicists with no prior knowledge of the ACP system in a little over two working days.

The performance of the system is limited because of the small number of nodes and the fact that the nodes are low-speed 16 bit processors. However, two important aspects of the test bed system performance that can be investigated are the efficiency of utilization of the nodes and the possibility of bus contention on the global bus. The first issue was checked with a typical reconstruction code by evaluating the fraction of time the individual nodes spend executing user programs compared to the time they spend waiting for events from the host. In all cases this was greater than 90%, and could be made to approach 100% by having the user software double buffer events. The second issue was checked by comparing the performance of the system with all six nodes running to the performance with a single node running. Six times the performance of an individual node was obtained. Similar tests will be carried out on the Phase II system in 1985.

Full-scale Production Systems

The first full-scale production system is scheduled to be operational in summer, 1985. It will consist of at least 128 nodes using full-speed 32 bit microprocessors of at least two types (Motorola 68020 and DEC MicroVAX are the leading candidates at the moment). At the crate level it will use the high-speed 32 bit bus most appropriate for the processor in use in that crate. Clearly, VME/VMX is appropriate for the 68020. Either MULTIBUS II or VME/VMX may be suitable for other processors.

The crates, CPU nodes, and memories in this Phase II system are simply higher speed versions of the existing Phase I components in the test bed system. However, the crate interconnections are necessarily more complex to allow the use of a larger number of nodes. A tree-like system (see Figure 4) will be designed for the Phase II system. The host CPU functions, including I/O and system control, are in the root. The node crates are connected by simple, ACP designed, high-speed branch busses. These multiple branch busses, capable of operating simultaneously at 20 Mbytes/sec each, are interconnected via a bus switch which allows any one of several root masters to be connected to any one of the branches. This will support the highest performance requirements of future data storage devices and on-line high level trigger applications with

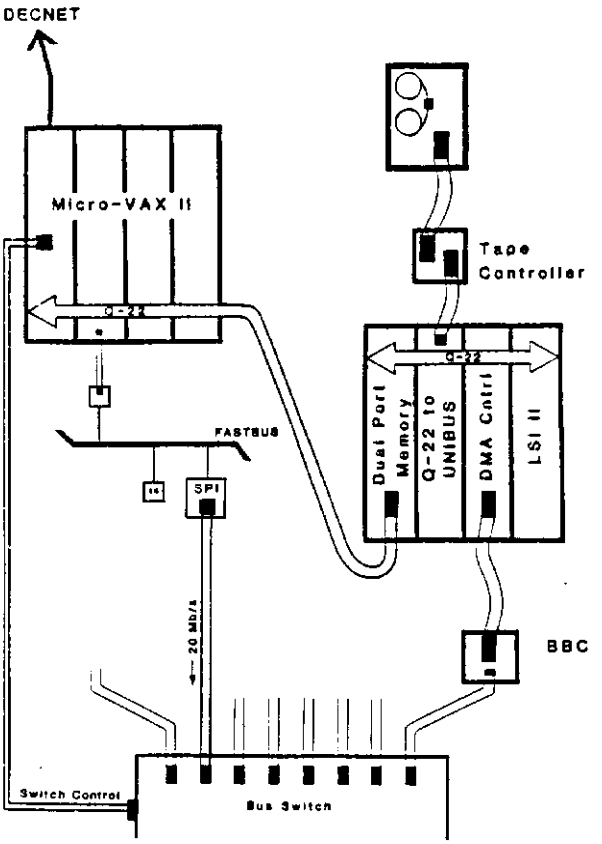

Figure 4. Branches and bus switch of Phase II system.

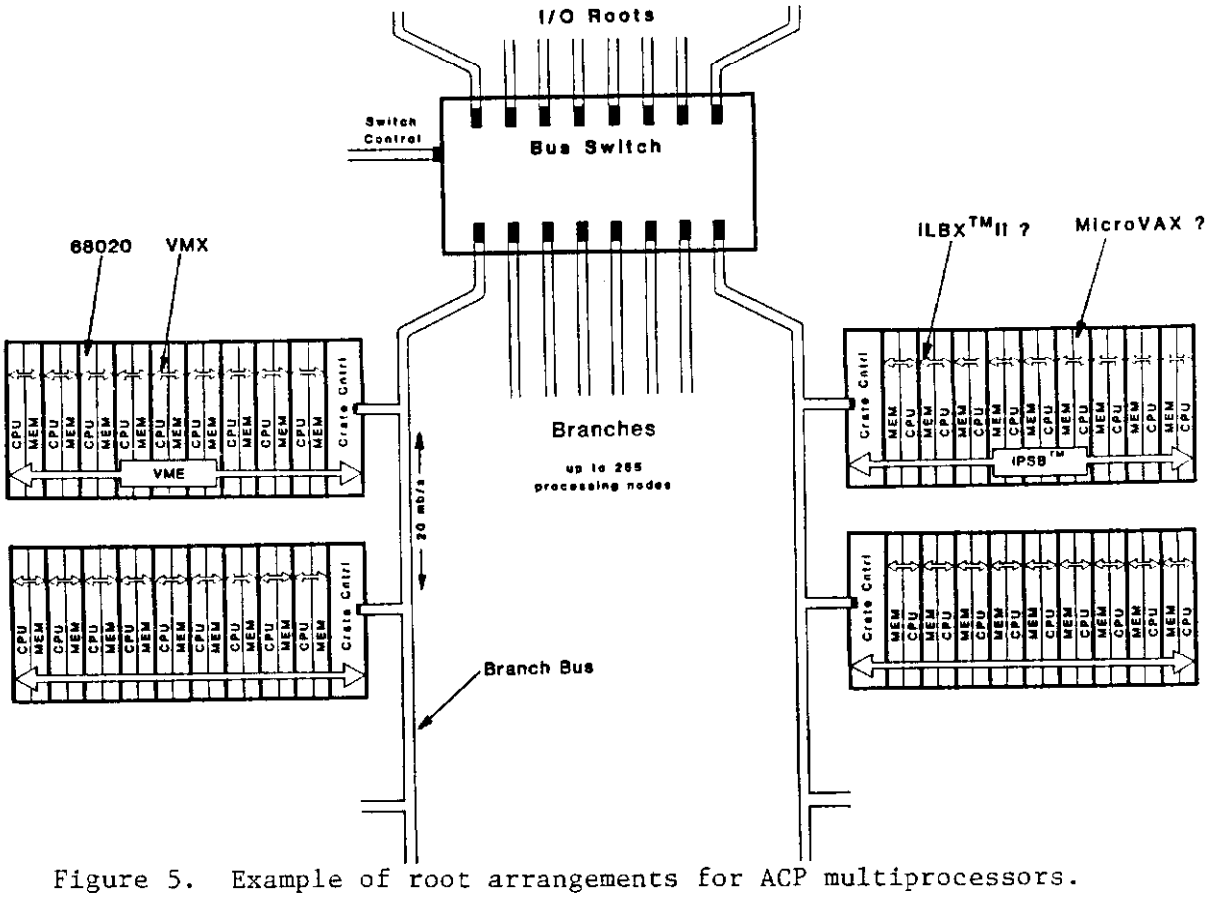

Figure 5. Example of root arrangements for ACP multiprocessors.

over 100 Mbytes/sec data bandwidth capability. The only overhead for this powerful multiple branch system is the relatively low cost of one bus switch, presently estimated at less than $5000. For the simplest implementations, with a limited number of nodes, a single branch bus can be used to connect all crates of nodes with a simple host interface as in the Phase I system.

As soon as the more pressing design issues at the node and branch level will allow, the single host CPU of the Phase I design will be replaced by a sophisticated root with a group of individual CPUs each performing a separate function (see Figure 5). LSI 11s (or similar devices) sitting on a Q-22 bus will act as input and output controllers for tape (or disk) operations. Each is connected through a UNIBUS converter to one or more tape drives and disks. They will operate under the familiar RT11 system to pass data between tape and the nodes through a Q-bus DMA I/O device and a branch bus controller (BBC), the master on the branch bus.

The user's production host software will run on a separate CPU, most likely a MicroVAX running MicroVMS. This CPU sits on a second (global) Q-bus. A memory with two Q-bus ports services its local LSI 11 and the MicroVAX. This allows the user high level software in the MicroVAX to initiate execution of the I/O and node communication primitives in the LSI 11. System control software monitors the status of the nodes, sets the bus switch, and transmits the node address cycle before each block of data cycles. This software also resides in the MicroVAX which is connected to the switch control port.

Also shown in Figure 5 is a root connected to a FASTBUS on-line data acquisition system through a special processor interface module (SPI), which is a master on the branch bus. In this environment, the host MicroVAX is informed by FASTBUS of a ready event and its type. The MicroVAX, under control of user software, sets the switch and transmits the node address just as it does when operating with a tape drive as described above. It then instructs the FASTBUS system to transmit the event over the appropriate root channel. The bus switch can support up to eight such root channels operating concurrently, each carrying up to 20 Mbytes/sec. This can include one or more FASTBUS channels, along with tape or disk I/O channels. This flexible and modular root system provides a cost-effective implementation of host CPU functions for off-line systems, as well as a convenient way to use the same collection of nodes with unchanged user software in both on-line and off-line environments.

In some sense, this has been a description of a Phase 2.1 system since, as already alluded to, the ACP may not have the design resources to develop the components of the root which are not commercially available on the time scale planned for Phase II. Early testing of the first full scale system may take place using a single rather than double Q-bus system, or even a VAX 11/780 as the production host much as has been done for the test bed system. However, the latter configuration would only take advantage of about half of the full data rate capabilities of 6250 bpi tape drives. For this reason, because of the large cost savings, and because of its importance in on-line activities a multiple micro-CPU root will be brought on-line as early as possible.

Conclusion and Future Directions

Phase III, starting in the second half of 1985, will build on the modules developed in Phase II to provide higher performance and more specialized versions of the ACP hardware. This will include implementing the production

host with more cost-effective processors, the incorporation of higher performance nodes, and the development of special purpose hardware coprocessors for a variety of particular algorithms. The flexible bus switch, and a nearest neighbor connection module which may be developed in Phase III, will be exploited to provide more complex node interconnection schemes in both grid-like and multiple rank systems.

A large amount of industry effort, including both minicomputer and semiconductor manufacturers, is converging in the direction of making VAX class VLSI products available at the chip and board level. The ACP is developing the hardware and software structure to take early advantage of this most cost-effective and flexible solution to high energy physics production computing needs. It has demonstrated user support software that makes it relatively comfortable for physicists to take advantage of multiprocessing. In the course of these activities, the ACP is testing multiprocessor architectures and solving system problems, both in hardware and software, that are relevant to many computer research activities outside of high energy physics.

References

1. Thomas Nash, et al. "Fermilab's Advanced Computer Research and Development Program," Proceedings, Three Day In-Depth Review on the Impact of Specialized Processors in Elementary Particle Physics, Padova, 1983, p.227.

2. See other papers in the Proceedings of this conference as well as Proceedings of Topical Conference on the Application of Microprocessors to High Energy Physics Experiments, CERN, Geneva, Switzerland, May 4-6, 1981 (CERN 81-07), and Proceedings of Three Day In-Depth Review on the Impact of Specialized Processors in Elementary Particle Physics, Padova, Italy, March 23-25, 1983.

3. Hari Areti, et al. "ACP Modular Processing System: Design Specifications," Rev. April 2, 1984, FN-402.

4. Mark Fischler, et al. "Software for Event Oriented Processing on Multiprocessor Systems," Proceedings, this conference, FERMILAB-Conf-84/64.

5. Mark Fischler and Thomas Nash, "Computing Tools for Accelerator Design Calculations," Report of DPP Workshop, Accelerator Physics Issues for a Superconducting Super Collider, Ann Arbor, December 12-17, 1983, UM HE 84-1, page 113.

6. Advanced Computer Program, "ACP Software User's Guide for Event Oriented Processing," Rev. June 18, 1984, FN-403.