Fermi National Accelerator Laboratory

FERMILAB-Conf-88/097

High Performance Parallel Computers for Science: New Developments at the Fermilab Advanced Computer Program*

T. Nash, H. Areti, R. Atac, J. Biel, A. Cook, J. Deppe, M. Edel, M. Fischler, I. Gaines, R. Hance, D. Husby, M. Isely, E. Miranda, M. Miranda[a], T. Pham, and T. Zmuda
Advanced Computer Program
Fermi National Accelerator Laboratory
P.O. Box 500, Batavia, Illinois 60510

E. Eichten, G. Hockney, P. Mackenzie, H. B. Thacker, and D. Toussaint[b]
Theoretical Physics Group
Fermi National Accelerator Laboratory
P.O. Box 500, Batavia, Illinois 60510

August 1988

*Talk given by Thomas Nash at the Workshop on Computational Atomic and Nuclear Physics at One Gigaflop, Oak Ridge, TN, April 14-16, 1988.

Operated by Universities Research Association Inc. under contract with the United States Department of Energy

High Performance Parallel Computers for Science: New Developments at the Fermilab Advanced Computer Program*

T. Nash, H. Areti, R. Atac, J. Biel, A. Cook, J. Deppe, M. Edel, M. Fischler, I. Gaines, R.Hance, D. Husby, M. Isely, E. Miranda, M. Miranda[a], T. Pham, and T. Zmuda

Advanced Computer Program
Fermi National Accelerator Laboratory†
Batavia, IL 60510 USA

E. Eichten, G. Hockney, P. Mackenzie, H. B. Thacker and D. Toussaint[b]

Theoretical Physics Group
Fermi National Accelerator Laboratory†
Batavia, IL 60510 USA

ABSTRACT

Fermilab's Advanced Computer Program (ACP) has been developing highly cost effective, yet practical, parallel computers for high energy physics since 1984. The ACP's latest developments are proceeding in two directions. A Second Generation ACP Multiprocessor System for experiments will include $3500 RISC processors each with performance over 15 VAX MIPS . To support such high performance, the new system allows parallel I/O , parallel interprocess communication, and parallel host processes. The ACP Multi-Array Processor, has been developed for theoretical physics. Each $4000 node is a FORTRAN or C programmable pipelined 20 MFlops (peak), 10 MByte single board computer. These are plugged into a 16 port crossbar switch crate which handles both inter and intra crate communication. The crates are connected in a hypercube. Site oriented applications like lattice gauge theory are supported by system software called CANOPY, which makes the hardware virtually transparent to users. A 256 node, 5 GFlop, system is under construction.

---

* Talk given by Thomas Nash at the Workshop on *Computational Atomic and Nuclear Physics at One Gigaflop*, Oak Ridge, TN, April 14-16, 1988.

† Fermilab is operated by Universities Research Association, Inc. under contract with the U.S. Department of Energy.

1. INTRODUCTION

High energy physicists, like most scientists, have always wanted more computer power than they could afford to buy.[1] In the commercial marketplace, the emphasis is on software backward compatibility, product differentiation, and isolation of a client "herd". On the scientist's computing agenda, raw processing power in a relatively easy to use form, not corporate profitability, is the dominant issue.

Despite the fact that industry is not motivated by the rest of the market to provide the extremely cost effective computer *systems* demanded by much of science, it does provide an extraordinary array of *components* (chips, modules, peripherals, work stations, software, etc.) that can be assembled into what science requires. For over 5 years the Advanced Computer Program (ACP) at Fermilab has drawn from industrial components to design and produce usable parallel computer systems of such cost effectiveness that high energy physics (HEP) experiments are now being carried out that would otherwise be unthinkable.

In addition to its dependence on the latest offerings of industry, the ACP also takes advantage of developments coming from computer science. However, just as with industry, the motivations of natural and computer scientists have significant areas of divergence. When commissioning a new architecture prototype the interest of computer science primarily extends to proof of principle. Although also concerned with the latest from computer science, the ACP has the goal driven motivations of its scientific clientele. The ACP's activities, therefore, fall in the rather wide gap between the exploratory spirit of computer science and the production reliability and conservatism of industry. ACP systems must both take advantage of advanced concepts and components and be robust enough to satisfy a large community of demanding users.

In common with computer science, the ACP is aiming toward reasonably general purpose parallel computers. The difference is that the ACP cannot focus, as computer science usually does, on the often academic complications of the end goal. Rather, the ACP must take a step wise approach, dealing with the easier, more obviously parallel, problems first. At each step as more complex issues are addressed, there must be a fully working, commercializable machine that delivers the maximum cost effectiveness available at the time.

With their different perspective, natural scientists have brought to the computer development community new philosophical and architectural approaches to investigate. The dynamic nature of scientific problems means that programs are frequently changed, often extensively. The scientific user generally has a deep understanding of the structure of the problem and the capability and willingness to map the algorithm to optimally cost effective computer architectures. Issues of software compatibility can certainly become painfully acute at mundane levels like compiler syntax. However, when it comes to the global scale of a problem, scientists are virtually unique at this time in their receptiveness to explicit (user directed), as opposed to implicit (automatic), decomposition of programs onto parallel computers.

Fox and Seitz at Cal Tech, driven initially by the needs of theoretical physics, established the viability of hypercubes, the quintessential explicitly parallel, local memory architecture.[2] These are now the subject of much attention in computer science which earlier had focussed heavily on global memory architectures and attempts at automatic parallelization of whole algorithms. Similarly, the natural parallelism inherent in the experimental HEP computing problem (running the identical reconstruction program on many millions of different events) suggests a simple parallel processing solution. The pioneering emulator work by Kunz *et al.* at SLAC demonstrated the feasibility of using multiprocessor systems to provide cost effective computing for the reconstruction of experiment events.[3]

Almost a decade after the first SLAC emulator, powerful 32-bit microprocessors allowed the ACP at Fermilab to develop even more convenient and cost effective event oriented parallel processing systems using many more CPUs.[4] Such systems are now an acknowledged important component of computing in high energy physics. They have made crucial contributions to data analysis in a number of current experiments, and will be indispensable for the large experiments of the SSC era. The first generation ACP systems are rapidly becoming a standard. There are over 30 installations in universities and laboratories worldwide, primarily, though not exclusively, for HEP applications. The first of these ACP systems, which provide CPU power at a cost of less than $2500 per VAX 780 equivalent, were brought on line two years ago. The initial 110 processor system, along with several newer installations, have been heavily used at Fermilab for physics event reconstruction.

This paper will describe new work from the ACP in two areas: a second generation multiprocessor targeted at experiment applications, and a multi array processor "supercomputer" for the site oriented problems of theoretical physics, principally lattice gauge calculations. The mandate for ACP development remains the computing needs of high energy physics. However, the potential applicability of these new systems, particularly the one aimed at theoretical physics, is far broader than HEP alone.

We will first cover the new ACP systems for experimental HEP which, early in the next year, will provide well over an order of magnitude increase in cost effectiveness over the original systems. This second generation project will also allow much higher bandwidth for both I/O and interprocessor communication, and will have software tools allowing almost any UNIX , VMS , or (potentially) VM based processor to be used equivalently as a node or "front end" in a multiprocessor ACP system. Perhaps most important is the way in which this new system allows for integration of powerful "back end" multiprocessors, now usable for both reconstruction and physics analysis, into a modern Ethernet based workstation environment.

The multi array processor system for theory will be described next. Programmable in C and FORTRAN and supported by a user interface that makes transparent the details of the hardware architecture, this system is also at the highest levels of cost effectiveness. A 5 GigaFlop (peak), 2.5 GByte, 256 node, system is expected to be running in early 1989 at a cost of approximately $1 million. This system, like all others from the ACP, is intended for quick commercialization so that it will be available to other institutions with needs similar to Fermilab. At the end of the paper, we will outline expectations of future directions ACP developments may take including the likely merger of the experimental and theoretical machines.

2. THE SECOND GENERATION ACP MULTIPROCESSOR

The continued saturation of computers (including ACP systems) by HEP experimenters motivates the development of a second generation of the successful ACP Multiprocessor system. Industry is continuing to provide a stream of often surprisingly powerful technology that we can harness to meet the severe challenges of this science. Frankly, some of industry's new components are making our task particularly pleasurable and rewarding these days. A variety of new and increasingly powerful microprocessors are now available to incorporate into multiprocessors. Beyond the new versions of existing processors (like the 25 MHz 68030s replacing 16 MHz 68020s), there are entirely new families of chips. Most of these are based on the Reduced Instruction Set Computer (RISC) architectural philosophy. It was noted in the 1970s that many complex instructions in traditional machines with large instruction sets, like the IBM 360s and DEC

VAXes, were rarely used. They effectively increased the cycle time for all instructions because they mandated extensive microcode. RISC machines are generally pipelined with no microcode and very fast instruction cycles. They defer complex instructions to software.

New processors that will offer at least a factor of three, and in one case as much as a factor of twenty, more performance than the first generation ACP processors based on 68020s include: the MIPS R2000 and R3000 RISC chips; the Fairchild, now Intergraph, Clipper chip set; the AMD 29000 RISC chip; the Sun SPARC RISC chip; the INMOS T800 transputer; the Motorola 88000 RISC chip; the National Semiconductor 32532 processor. Unlike earlier processors, many of these have usable FORTRAN and C COMPILERS and UNIX operating systems even before the hardware is available.

The broad availability of UNIX is particularly important. As we will describe below, the new ACP architecture will support any processor running UNIX (or VMS) that can be connected via VME or Ethernet. It is very difficult to predict with any certainty which processor or commercial single board (or single slot) computer (SBC) will be most cost effective in the future. The openness of the ACP system permits competitive purchase of processor nodes based on performance benchmarks and price.

The large increase in CPU power available for the Second Generation ACP System requires a redesign of the multiprocessor hardware and software system architecture to remove bottlenecks in current systems that would be felt at the higher performance levels. The bottlenecks are I/O and interprocessor communication bandwidth, and the CPU power available for the host process. Continuing the successful strategy of attacking computing limitations with parallelism, the new architecture solves I/O and host limitations by supporting parallel I/O and parallel host processing. Moreover, to avoid mini computer bus bandwidth restrictions, any node may take on host functions including I/O through controllers in its own local crate. Though significant, the changes in the new system hardware and software are designed to be as transparent as possible to users of the original system. In the following, we describe the fast new ACP SBC, and then the new interconnection modules and topologies, the new (parallel) I/O options, and the new software system architecture that are all designed to support the large increase in performance that will be available.

2.1. A Fast New ACP Processor Node

As it has in the past, the ACP is encouraging competition in SBCs targeted at parallel processing by developing an extremely cost effective VME SBC. The choice of the R3000 RISC processor from MIPS Computer Systems, Inc. for this design was based on performance evaluations of the microprocessors referred to earlier. The ultimate standard is how real physics code runs in a high level language, since it does not matter how many million instructions per second (MIPS) the CPU can execute if the instructions are not useful in FORTRAN or C and if the compilers fail to provide sufficient optimization. The ACP has performed benchmarks on several of the new chips using a suite of high energy physics FORTRAN programs (a small Monte Carlo/track reconstruction package; an actual fixed target tracking code running on experimental data; and a floating point intensive theoretical calculation). These benchmarks were run on a 16 MHz MIPS R2000 system. For the three benchmarks, performances were 7.9, 6.4, and 7.4 times that of a VAX 780, which is generally accepted as the standard 1 MIP normalization. The new ACP node will use an R3000, the new improved version of the R2000. Carefully taking into account processor, cache, as well as clock speed differences, leads to a projected HEP code performance of 12-15 VAXes for 25 MHz R3000 boards and 16-20 VAXes for the 33 MHz versions we expect to use.

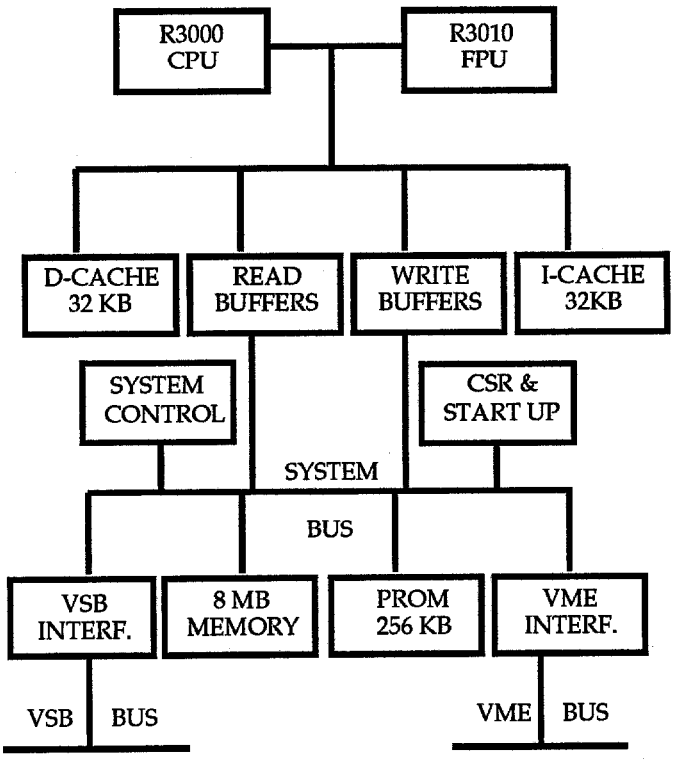

Figure 1. The ACP MIPS processor, block diagram.

The ACP MIPS processor module (Figure 1) consists of:

1) a 25 or 33 MHz MIPS R3000 CPU;
2) a 25 or 33 MHz MIPS R3010 floating point unit;
3) Four Write Buffers;
4) a 32 KByte instruction cache;
5) a 32 KByte data cache;
6) 8 MBytes of parity checking main memory, made up of 1 Mbit (100 nsec nibble mode access) DRAMs, expandable to 24 MBytes via VSB (VME System Bus);
7) interval timers that can interrupt the CPU;
8) a full function VME Master Slave interface supporting 20 MBytes per sec block transfers;
9) a full function VSB interface for peripherals and memory extensions.

The ACPMIPS processor module will be packaged as two connected boards that will fit in one standard VME slot. One board will contain the processors and cache; the other board will have main memory and the VME and VSB interfaces. Up to two extra memory boards may be plugged into other slots and connected via VSB to allow a total of 24 MBytes. The modular nature of this design means that as new components (denser memory or faster processors) become available, only one board at a time needs updating. With 4 Mbit DRAM, expected in a year or so, 32 MBytes will be possible without an expansion board (or up to 96 MBytes with maximum expansion).

The new CPU module will provide high level language processing power with a cost effectiveness of well under $200/VAX 780 equivalent, based on the physics benchmarks and features listed above. The FORTRAN compiler is the best we have encountered for a microprocessor. It supports VMS extensions and compares favorably with the VMS compiler in convenience and sophistication. Since the MIPS CPU chip has on chip memory management, the board will be able to run the full UNIX operating system, booting either from a VME disk drive or using the Network File System (NFS) over the Branchbus. Full UNIX program development tools are available. This processor will form the cornerstone of the second generation ACP systems.

2.2. New Interconnect Topologies

The original ACP multiprocessor used a single (MicroVAX) host which was the master of large numbers of microprocessor nodes. This system has been extensively described in earlier papers.[4] Here we discuss only the significant new improvements. The ACP Branchbus was developed to link several high performance commercial local bus crates (like VME) to a host and/or a data acquisition system. It is optimized for high speed block transfers.[5] No commercial

alternative was available. The original Branchbus is a 32-bit bus connecting a single master (QBus, Unibus or FASTBUS) to multiple crates (VME), with block transfers at up to 20 MBytes/sec. Improvements to the Branchbus system of interfaces allow higher performance and more complex interconnection schemes.

The new VME Branchbus Controller (VBBC) allows any VME master, in particular any node (or smart I/O device controller) in an ACP multiprocessor system, to act as a Branchbus master and read or write to any Branchbus address. Any processor in the system can communicate with any other processor without host intervention, allowing the more elegant system architectures described below. Figure 2 shows a block diagram of a high performance off-line ACP system using the VBBC. Here no specialized MicroVAX is required as a host; a standard node takes on that function, and an accompanying smart tape controller can transfer data directly from tape to any node in the system. The VBBC is a VME slave and Branchbus master. It is a shared resource in the VME crate, allocated on a first-come first-served basis by a test-and-set bit in its control register. Once programmed with the Branchbus control words and address, the Branchbus cycles occur transparently to the VME master which simply reads or writes data to the VBBC. First prototypes of the VBBC are now working.

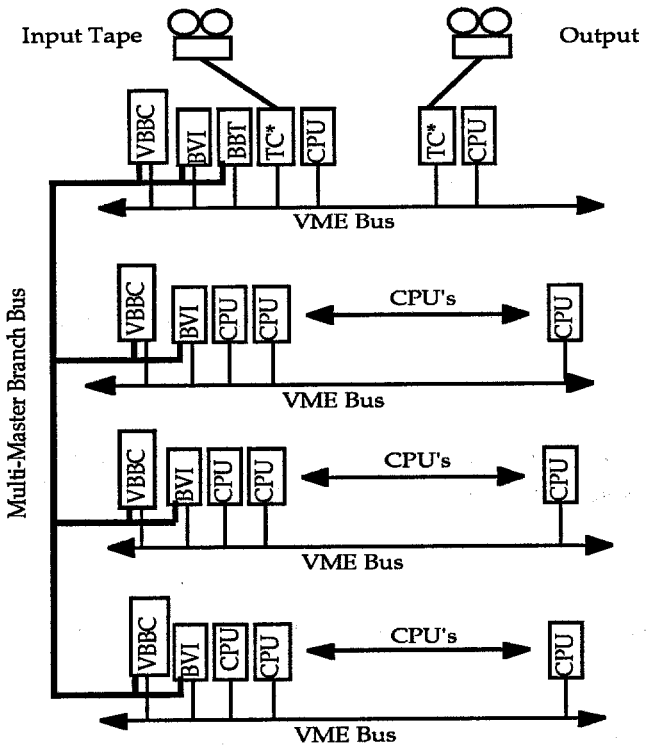

Figure 2. A High Performance Off-Line ACP System

As implied above, the Branchbus now supports multiple masters. A distributed arbitration scheme similar to that used on the SCSI bus allows up to 16 masters to share the bus. Existing masters (Q/UBBC modules) can be used in multi-master systems by replacing their existing Branchbus Interface Daughter Board with a new Multi-Master Branchbus Interface Daughter Board which handles the arbitration transparently to the user.

The ACP Branchbus Switch allows full crossbar interconnection of up to 16 Branchbuses (or more using multiple switches). With this switch, any Branchbus master device can connect to any slave in the entire switch connected system. All channels of the switch can be active simultaneously. For example, eight of the Branchbuses could be connected to the other eight, all transferring data simultaneously giving an aggregate bandwidth of 8 X 20 MBytes/sec or 160 MBytes/sec (in addition to any local bus activity on any of the VME crates in the system). Thirteen Texas Instrument 16 X 16 X 4 bit crossbar chips (TI 74AS8840)are used for the main switching elements. The Switch is a backplane in a 6U by 280 mm Eurocard crate. Modules may be plugged into the Switch Crate (see Figure 3) much as with VME. However, instead of the signals being connected in a bus structure, each slot in the crate is a crossbar switch point. The first two Switch crates are built and working.

Two modules now exist that plug into the Switch Crate. One is the Branchbus Switch Interface Board (BSIB) which converts the differential RS485 signals on the standard Branchbus ca-

bles to the single ended TTL version of Branchbus used on the Switch backplane. The BSIB brings one standard Branchbus, with its VME crates, host, Fastbus, etc., into a port of the Switch, and allows them to be switched to whatever else is plugged into the Switch Crate, such as other Branchbus circuits. Use of the Switch in this way will allow multiprocessor systems to obtain extremely high bandwidth for interprocessor communication. The second existing module that plugs into the Switch is the Floating Point Array Processor (FPAP), a 20 MFlop device, that is used for theoretical physics, primarily lattice gauge, calculations. The FPAP is described later in detail, as is the innovative "better than a hypercube" architecture that the Switch allows. Important future applications of the versatile Switch are described in the Section 4.

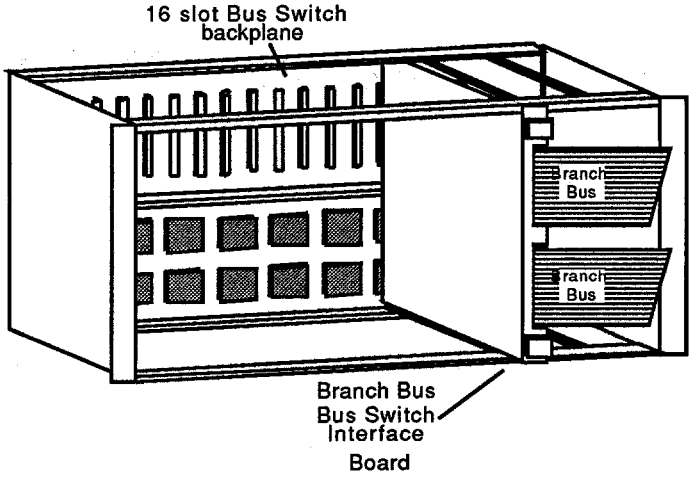

Figure 3. The ACP Branchbus Switch. The backplane uses single ended TTL Branchbus protocol.

2.3. Parallel Input/Output

Although it will continue to be possible to read and write tapes through a VAX or MicroVAX into an ACP system, high performance operation will take advantage of I/O devices that interface directly to the multiprocessor system bus. Unlike Unibus and QBus tape drives, VME interfaces will give a bandwidth potential, depending on the I/O device, of up to the 20-30 MBytes/sec allowed by VME rather than the roughly 1 MByte/sec possible with minicomputer buses. One such VME tape interface, the Ciprico TM3000, has been used for data acquisition at Fermilab.

Parallel I/O is particularly encouraged by the availability of inexpensive, high capacity mass storage devices which interface directly to VME. It is made possible by the availability of I/O directly in the node crates and by the new capability of a processor and/or its intelligent I/O controller to write through a VBBC from one crate to another. The multiprocessor system will have available to it many devices all reading and writing simultaneously, allowing the total I/O bandwidth to be increased to whatever level is required.

New I/O devices are replacing standard 6250 bpi magnetic tape. These are very appealing to HEP experiments anticipating huge amounts of data, such as one approved Fermilab experiment which is planning to record tens of billions of events. At this time video tape cassettes appear most promising. The 8 mm format can pack well over 100 times the data in a given volume of shelf space as can conventional tapes, and they are at least an order of magnitude more cost effective. Current devices allow bandwidths comparable to those achieved with standard 6250 bpi tapes, with further improvements expected in the next year.

The ACP has tested the 8mm video tape device manufactured by Exabyte Corporation. Currently available devices store 2 Gbytes (as much as 12 conventional tapes) on a standard $10 tape cassette. The drives cost less than $3000 and can deliver data at 250 KBytes/sec through a SCSI bus interface to VME or QBus. Both density and speed are expected to double in the next year.

The Exabyte drive package allows the design of a cheap and simple mechanical loader. Such a design is underway. It will incorporate a reader for optical bar code labels to insure mounting of the correct set of tapes from a large data sample. Automating tape loads eliminates the time for human operators to mount tapes which becomes significant when tapes are being

processed at high speed.

The ACP plan at this writing is to support the Exabyte drives. However, it is important to emphasize that no standard has yet emerged in this rapidly developing field, and the long-term reliability of such devices has not been established. Nonetheless, it is clear that one can count on far better I/O performance than has been available.

2.4. Software System Architecture

The redesign of the system software for the Second Generation ACP System will support the greatly improved performance and flexibility of the new processing, I/O, and communication hardware, while reducing the complexity encountered by both beginning and sophisticated users.[6] It will allow existing applications to run with minimal changes, yet will provide a variety of powerful new features to allow users to realize the full potential of the new processors. Integration will be possible into a variety of computing situations, including large mainframe computer centers and the traditional VAX or MicroVAX host. Most important, in our view, is a distributed computing UNIX (or VMS) workstation environment in which the multiprocessor will function as a fully integrated back end engine directly controlled from the workstations. The general hardware environment supported is shown in Figure 4. Because of the emphasis on portable, nearly universal standards, it will be straightforward to incorporate any computer that runs UNIX and/or can communicate via TCP/IP over Ethernet. This makes the system open and receptive to future requirements and product options.

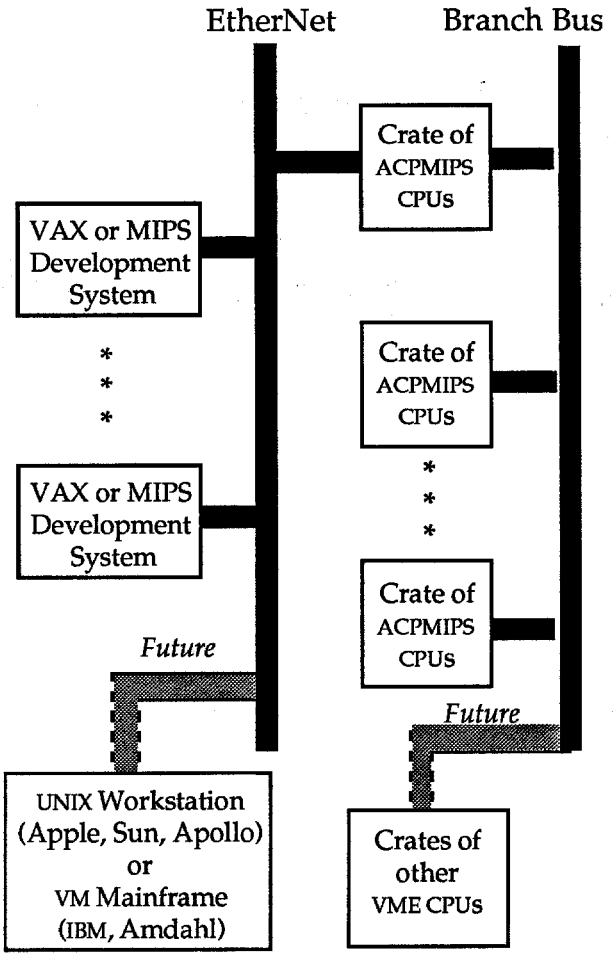

Figure 4. Second GenerationACP System: General hardware and network environment.

As before, the ACP System Software is a tool to make it easy for physicists to bring their programs to a high performance multiprocessor environment. An application for the second generation system is decomposed into a set of cooperating processes. Support for running these sets of processes as a job, including interprocess communication and synchronization, startup, etc., is provided by the ACP System Software. The processes run on an ACP Multiprocessor System interconnected by Branchbus and on any associated UNIX or VMS (later possibly also VM) computers connected via Ethernet and TCP/IP. They may be distributed as appropriate over the available computers. In this way a multi process job may be tested first in a single machine and then with increasing numbers of nodes. Program development is done using the full set of UNIX (or VMS) tools, including compilers, linkers and de-

buggers, of the computer on which the process will run.

Any node process can assume the functions previously exercised only by the VAX host processor. Any node process in the system can do I/O, reading or writing data tapes and accessing disk files. And any node process in the system can do `send` or `get` operations to or from an individual process (chosen by the system software from a class or rank of node processes) or set of processes in a given class or rank. As before, the system software will automatically find an available node process for the user. The ability to `send` and `get` to multiple nodes in a class allows `broadcast` and `accumulate` type operations.

Along with the traditional `send` and `get` type of ACP communication routines, there will be a variety of more primitive, yet powerful and easy to use, interprocess communication mechanisms. A process may send a block of data directly to a block of virtual address space in another process, or it may call a subroutine in another process (remote subroutine call), or it may send a small data packet (a message) to another process. Users of the new system will have direct access to process queues which they may define as they require or use in standard, traditional ACP defined ways (like node process *ready* or *complete*). Synch points provide a way for all processes in a class to synchronize program execution.

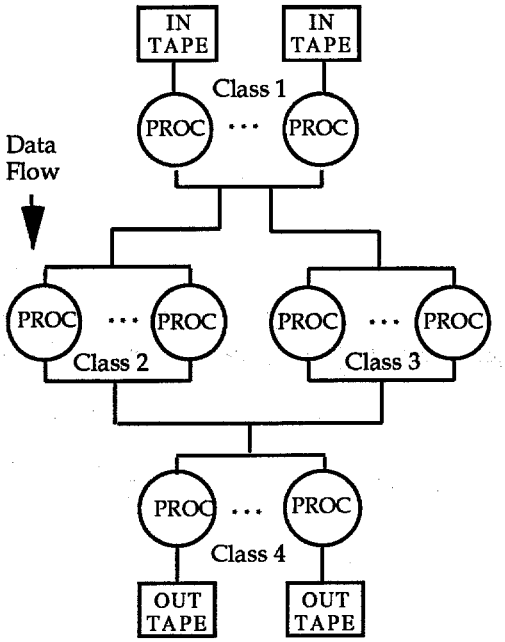

Figure 5. A multi rank configuration. Processes are indicated by circles.

There are many possible process configurations that the new system can support. An example for a reconstruction problem with multiple input tapes is shown in Figure 5. Note that this is a software configuration; the actual hardware connection of the nodes is over Branchbus via VBBCs and Bus Switches (if necessary) and is transparent to the programmer. Nodes in the top rank read events from data tapes and pass them along to either class 2 or class 3 nodes, which process events of different trigger types. Nodes in the bottom rank collect events from any nodes in the middle rank, either class 2 or class 3, for output to tape. Another important configuration has enormous implications for physics data analysis (as opposed to reconstruction). Here, the top rank of nodes is the same as in Figure 5, and the second rank consists only of one or more workstations and, perhaps, a single data recording process. With such a configuration, a whole experiment's data base of data summary tapes (DSTs) can be read and analyzed in parallel in much less than an hour. The traditional means of passing hundreds of DSTs through a computer center for a physics analysis pass often takes weeks.

To summarize, the Second Generation ACP System architecture is designed to support large increases in processing power and in I/O and communication bandwidth. It provides a set of hardware and software building blocks so that a system can be matched to the set of applications it will run. One can provide enough I/O devices and Branchbus interconnects so there are no bandwidth limitations. The number of basic CPU nodes is determined by processing power requirements, while special computer nodes, such as workstations or mainframes, can be incorporated into the Ethernet side of the system when needed and available. As each job is run on the

system, nodes are assigned to run particular user processes (input, output, event processing, etc.) as appropriate. Not only the traditional compute bound event reconstruction, but also more I/O intensive data analysis will find a home on these systems.

3. THE ACP MULTI-ARRAY PROCESSOR SYSTEM FOR THEORISTS

The ACP Multi-Array Processor System (ACPMAPS) is a highly cost effective, local memory parallel computer designed for floating point intensive grid based problems. The project is a joint effort of the ACP and Fermilab's Theoretical Physics Group. Processing nodes of the system are single board array processors based on the FORTRAN and C programmable Weitek XL chip set. They are connected by a network of very high bandwidth 16 port crossbar switches. The architecture is designed to achieve the highest possible cost effectiveness while maintaining a high level of programmability. At Fermilab the primary application of the machine will be lattice gauge theory.

To obtain some estimates of the computing needs of lattice quantum chromodynamics (QCD) one can consider the calculation of the deconfining temperature in SU(3) gauge theory without quarks,[7] which is one of the most solid four dimensional calculations done so far. Something like 500,000 MFlops - hours (peak, ~70% delivered) were used on a Star ST-100 array processor. This calculation required a lattice spacing of less than 0.1 fermi and a volume of close to (2 fermi)^3, resulting in lattices with spatial sizes of up to 19^3. It is virtually certain that calculations with quarks will require even larger lattices than this for comparable accuracy. Lattices with space-time sizes 32^4 to 64^4, requiring 1-20 GBytes of data memory, are a reasonable guess. Calculations of hadron masses in the approximation of ignoring dynamical quarks have not yet achieved a reasonable understanding of calculation errors, even on Cray-sized supercomputers. Although algorithms for the inclusion of dynamical quark effects have made tremendous progress in the last few years, at present they still seem to require at least two orders of magnitude more computer time than comparable calculations without quarks. It is thus clear that large increases in combined CPU power *and* algorithmic power are still required even for simple QCD calculations.

The aim for the new ACPMAPS machine is to deliver such large amounts of memory and CPU power at the lowest possible cost, without compromising the programmability required for rapid algorithm development which is just as important as raw computing power in achieving the goals of lattice gauge and other problems in theory. Descriptions of the other large-scale computer projects aimed at lattice gauge theory may be found in the references[2,8].

A 16 node system is being built this summer. Two switch prototypes are working and tested. Four Floating Point Array Processor (FPAP) node modules are also working and undergoing rigorous testing. They have successfully run extensive physics code. Fermilab intends to proceed to a 256 node (5 GFlop for about $1 million) system as soon as the 16 node system is operational. Parts are being procured for this system which will be assembled at the end of the year. Given the communications bandwidth noted below and the large amount of memory per node, it has been calculated that performance for appropriate lattice gauge algorithms should increase linearly well past 256 nodes. Maximum system size is 2048 nodes. The system is being designed in the ACP tradition to be commercialized and available to other institutions.

3.1. Architecture Overview

A block diagram of the system is shown in Figure 6. The individual single board FPAPs have peak performance of 20 MFlops. Performance of key kernels (SU(3) multiplies) have been measured on the prototypes to exceed 15 MFlops/node. For lattice gauge physics, a performance standard is the link update time. Using the Kennedy-Pendleton heat bath algorithm (pure gauge) a single FPAP has been clocked at under 800 μsec per link update. Depending on the algorithm, this corresponds to a real performance of 4-10 MFlops/node. Each FPAP contains 8 MBytes of data and 2 MBytes of program memory. (In the event of memory shortages, the FPAPs can be configured with a minimum of 4 MBytes for data.)

The FPAPs are plugged into a crate whose backplane is a 16 fold bidirectional high speed crossbar. This is the Branchbus Switch Crate described earlier. The nodes can speak with each other in pairs at a full 20 MBytes/sec simultaneously. The architecture of ACPMAPS is a hypercube network of such crossbar switch crates each supporting 8-16 FPAPs. In a typical configuration 8 array processor nodes will be plugged into each switch crate along with up to 8 BSIB I/O modules (also described in an earlier section) that interconnect crates in a hypercube (or better, if extra interconnects are desired).

Processing nodes do not participate in any communication activity other than their own. This is an important distinction from traditional hypercube implementations. The switches handle intra and intercrate routing automatically. The system therefore does not operate with all

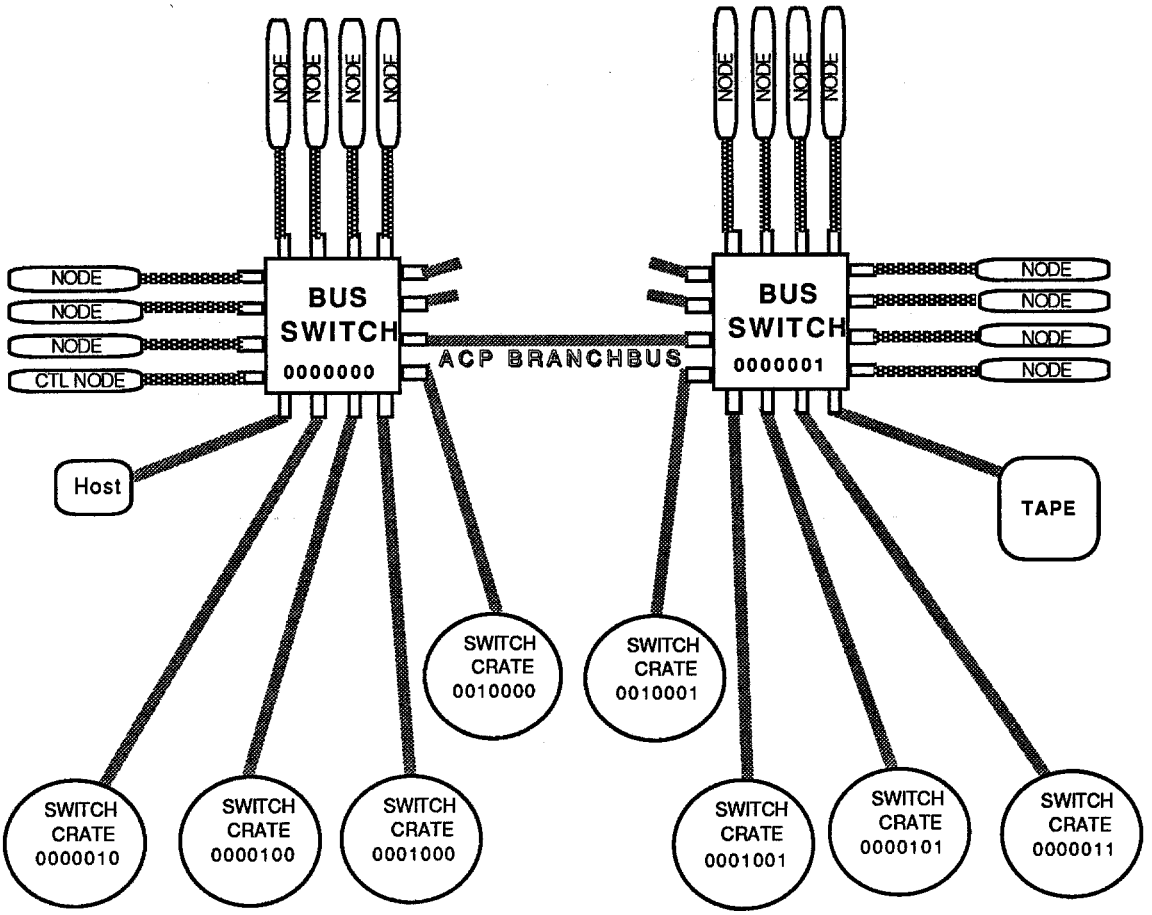

Figure 6. ACP Multi Array Processor System: 256 node configuration.

node programs (and/or communications) in lock step like an SIMD machine, as is the case in most of the other projects of this type[8] (Columbia, IBM GF11 and APE). It also does not strongly favor local communication (as existing hypercubes[2] do). It thus allows for any conceivable new lattice algorithm unconstrained by synchronous or local communication requirements. Despite its algorithmic flexibility the system ranks as the best (or nearly so, we won't argue) in terms of cost effectiveness of MFlops/$.

In addition to the flexible global architecture of the system, there are two important aspects of the FPAP itself that distinguish it from the CPUs of the other more special purpose lattice gauge processors. First, the FPAP memory is neither too large nor too small, but is ideally matched to the FPAP performance and communication capabilities and the demands of lattice gauge (and presumably other site oriented) problems. Secondly, the FPAP sticks closely to the Weitek XL architecture, using only one floating point processor per board. This allows the use of C and FORTRAN compilers and greatly simplifies the microcode.

The asynchronous communication and MIMD (multiple instruction, multiple data) processing architecture, in distinction to the more common synchronous communication, SIMD (single instruction, multiple data) approach, is one of the most important features of the system. There are many advantages to this type of architecture. It is very flexible: it can handle problems which are awkward or impossible in synchronous SIMD such as, in the case of lattice gauge, heat bath and incomplete LU decomposition algorithms and random lattice problems. The allowed sizes and shapes of the lattices are independent of the details of the hardware. The node structure of the machine can be made invisible in much or all of the high-level user code, resulting in improved programmability. This also results in improved fault tolerance, since the system can be reconfigured readily if a node fails, without requiring changes in user software or allocating nodes as spares. Complications which have to be faced include the potential for synchronization conflicts. This requires care in designing and understanding the communications system. In addition, a nontrivial system software design effort is required to ensure that overheads associated with the communications software are kept to acceptable levels.

A major new package of software (CANOPY) has been developed for this system. Theorist users need think only in terms of sites and fields on sites. The system automatically allocates sites to nodes and handles all site to site communication whether on the same node or another. Thus users do not have to know details of the hardware for effective use of the system. Routines that are used heavily will be microcoded. The skeleton of all applications are written in FORTRAN or C using a series of special subroutine calls that make the programs particularly readable for lattice gauge theorists and others with site oriented algorithms. In this way, despite ease of use and flexibility, the system can approach 10 MFlops/$4000 node in FORTRAN or C.* The CANOPY system software is described in more detail below.

3.2. The Floating Point Array Processor Module, Communication, and I/O

The initial ACPMAPS FPAP nodes are single board floating point array processors using the Weitek XL chip set which contains a 32 bit, 20 MFlop (peak) floating point unit, an integer processor, 32 floating point and 32 integer registers, and an instruction sequencer. The chip set as a whole is programmable in FORTRAN and C, at some sacrifice in performance. Thus, these modules incorporate the functions of a high level language programmable single board computer and a high performance floating point array processor. No external CPU is required as a controller for these stand alone floating point engines.

The FPAP modules (Figure 7) contain the XL chip set, the data and code memory, and the

* Memory prices are fluctuating widely at this writing. Costs given here are based on the (high) summer 1988 DRAM prices.

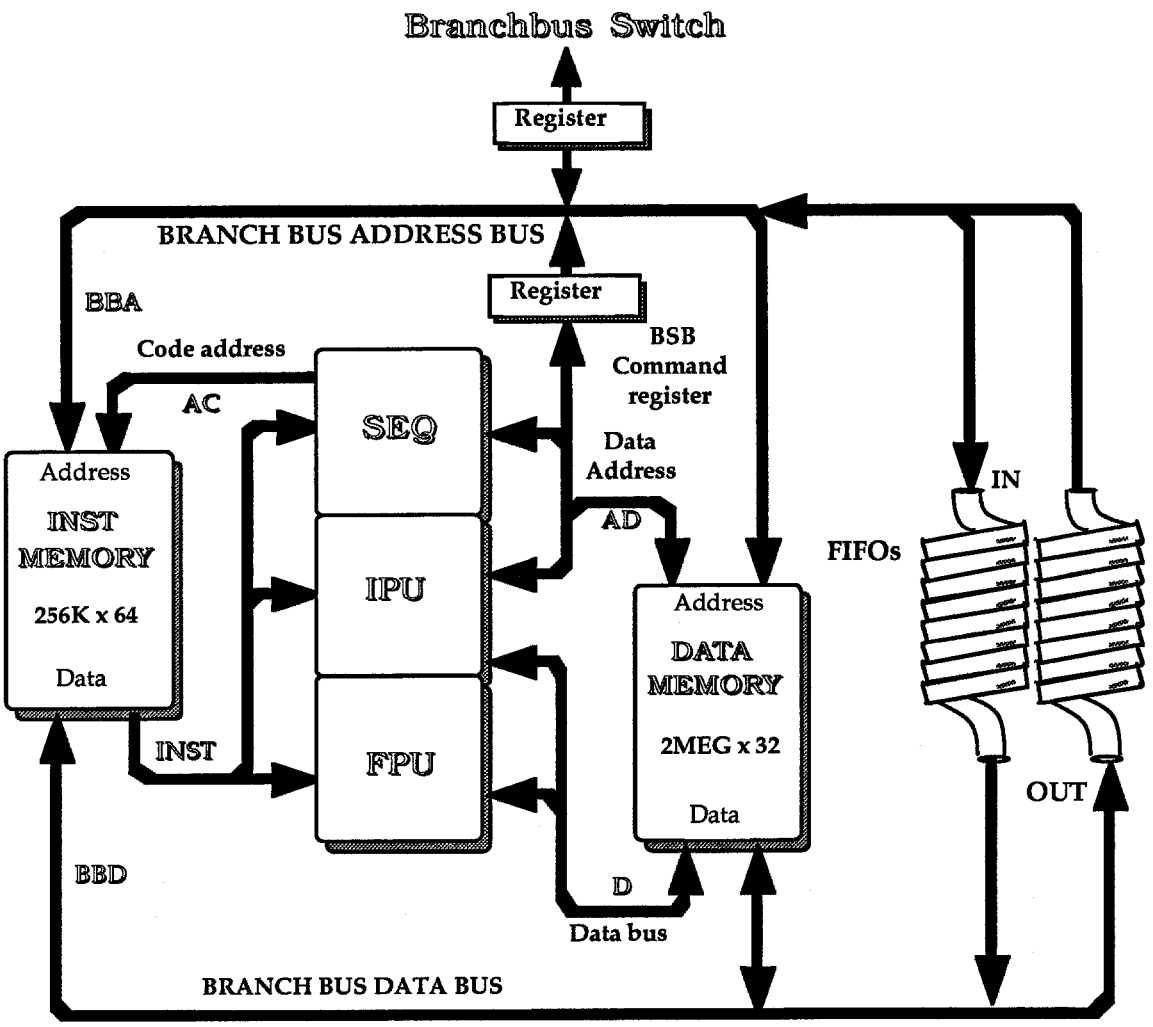

Figure 7. Schematic Design of the Floating Point Array Processor (FPAP)

interface logic and input and output queues for communicating with the crossbar switch crates. One floating point unit is used per node, in contrast to the designs of most of the other machines aimed at lattice gauge theory. In addition to being a flexible and sensible design for a wide variety of problems, this was dictated by the desire to be able to use the Weitek FORTRAN and C compilers for the XL chip set.

The 2 MBytes of program memory and 8 MBytes of data memory is made from 1 Mbit 80 nsec access time page (or static column) mode dynamic RAM chips. In page (or static column) mode these memory systems can deliver data at a rate of one word per 100 nsec. This rate is fast enough that little additional efficiency would be gained in most lattice algorithms by replacing some of the DRAM by faster, more expensive static RAM. The memory chips constitute at least a third of the total cost. The memory to power ratio provided (8 MBytes to 20 MFlops) is larger than that provided by most other machines of this type, and is larger than is required by presently existing algorithms for simulating full QCD including internal fermion loops. It is approximately appropriate for calculations in the valence approximation, ignoring fermion loops. Algorithmic improvements over the next few years will certainly change the required ratio. It seems likely that the possibilities which will increase the required ratio (preconditioning and Fourier acceleration of quark propagator calculation, Fourier acceleration of gauge simulation) are currently more promising than those which reduce the amount of memory required per CPU

cycle (such as adding nonlocal operators to the action to reduce finite lattice spacing errors) and that the large amount of memory could easily become crucial in the years to come.

The nodes are plugged into a network of Branchbus Switch Crates (described in Section 2.2) whose backplanes handle full sixteen port crossbar switching at bandwidths of 20 MBytes/second per connection. This yields a total bandwidth of 2.56 GBytes/sec for a 256 node machine. A cluster of 8-12 nodes is attached to each switch. The switches are connected in a hypercube, which may be augmented by additional communication channels along heavily used paths. This structure allows the nodes to communicate as if they were connected in a conventional hypercube arrangement, but more than this, it allows any node to communicate at full speed with *any* other node, allowing efficient running of algorithms requiring nonlocal communications. The Switch Crates allow any node to access any other node's data memory without needing to know where the other node is located on the network. With the current Switch Crate hardware, systems of up to 2048 nodes are possible before this transparent nonlocal communication feature is lost. The Branchbus Switch Crates will also be used in a variety of high performance experimental particle physics applications of the ACP Multiprocessor System, including the Level-3 programmable trigger for the Collider Detector at Fermilab, CDF.

The Exabyte video technology tape drives, described in an earlier section, will be used for check pointing long calculations and for archiving of gauge fields and propagators. One drive will be attached to every switch crate, enabling all of memory to be stored in under five minutes.

3.3. CANOPY System Software

Lattice gauge theories are part of a large class of grid-based problems derived from discretization of a set of differential equations which are very suitable for a parallel architecture like this one. The natural breakdown of the problem is to assign a certain subset of the sites in the space or spacetime to each node, which stores the data for the field variables defined on the sites assigned to it in its local memory and does calculations for its sites. The system software[9], known as CANOPY, has been designed to shield the user as much as possible from the hardware dependent node structure of the parallel architecture. The user thinks in terms of sites not nodes.

User programs are divided conceptually into two pieces: the control program, which is called from a MicroVAX host or mainframe VAX and runs on the control node, and site subroutines, which run on the individual nodes. The control program manages the execution of lattice-wide tasks. It is typically written in ordinary FORTRAN or C augmented by a set of system subroutines for dealing with global concepts (e.g., field memory, lattice wide tasks) which are distributed over all the nodes and require special treatment. The beginning of the control program includes statements like the following:

```
call define_periodic_lattice ( ndims, sizes, lat1 )
call define_field ( lat1, quarksize, q )
call define_field ( lat1, quarksize, q1 )
call complete_definitions
```

The routine `define_periodic_lattice` tells the system that our problem contains one lattice called `lat1` of `ndims` dimensions with the size of each dimension contained in the array `sizes` and with standard hypercube connectivity. More general user defined connectivity are allowed. It is possible to define several lattices in the same program for block spin renormalization group or multigrid algorithms. The routine `define_field` tells the system that memory will be required for storing two fields identified by `q` and `q1`, each with `quarksize` components for each site of `lat1`. The routine `complete_definitions` calls routines which assign specific sites to specific nodes, allocate memory in the nodes for the field data and site structures, and set

up structures for each site pointing to the memory areas of adjacent sites of the lattice.

A control node subroutine which operates on a field `q` with an operator `dslash_` and stores the result in another field `q1` would be written as follows.

```
subroutine dslash ( q, q1 )
        .
        .
        .
call do_task ( dslash_, lat1,
        pass$, q, 1,
        pass$, q1,1,
        end$)
return
end
```

The system subroutine `do_task` passes to all the nodes a pointer to a subroutine `dslash_` which operates on a single site and a pointer to a list of sites on which to operate, which may be the entire lattice `lat1` or some previously defined set of sites such as `red_sites`. A system routine on the node, invisible to the user, calls `dslash_` for all the sites in the set of sites which have been assigned to the node. `do_task` may also be used to pass (`pass$`) to the nodes parameters required by the site subroutine (like the field identifiers `q` and `q1`) and to integrate (`integrate$`) data returned from the individual nodes.

The site subroutines access and replace data from global fields with subroutines like:

```
call get_field  ( q,  site, qtemp )
call put_field  ( q1, site, qtemp )
```

which determine if the desired data is already resident on the node and open a channel to the communications hardware if necessary.

Most site subroutines can be written in FORTRAN or C. CPU intensive kernels such as SU(3) matrix multiplication and essential routines like `dslash_` will be microcoded for maximum efficiency. We expect that lattice gauge algorithms prepared in this way will run at up to 10 MFlops per node.

The main interest of the Fermilab lattice group in using CANOPY and the ACPMAPS system is the application of lattice gauge theory to QCD and "beyond the standard model" phenomenology. [10] However, since site oriented problems involving numerical solutions of differential equations pervade all of science and engineering, the hardware and software we have just described clearly have a much broader applicability than just high energy theory.

4. FUTURE DIRECTIONS

To a large extent future ACP directions, given the apparently insatiable demand of high energy physicists for computer cycles, will be driven by the extremely fast pace at which microprocessor performance is increasing. RISC processors running at clocks upwards of 50 MHz, with performance at 50 and even 100 MIPS now seem assured within only 2 years or so. Technically this puts a big demand on cache systems and on the speed of static RAMS used in caches. This problem must be solved by the processor companies since it is universal for their customers.

The other issue the extraordinary performance of these devices will raise in multiprocessor environments is the inadequacy of busses to support their communication requirements. It will not be a matter of picking a slightly faster bus than VME (like Multibus II or even Fastbus). The bus concept itself will be obsolete and will have to be replaced by point to point communication. For the ACP this appears to mean that future processors, even those for experimenters, will have to plug directly into the Branchbus Switch Crate. This will provide the communication bandwidth they need, but no longer in a commercial crate standard where the marketplace delivers a large and always improving variety of I/O controllers and other essential peripheral devices. The Branchbus Switch is the same height as VME but deeper. It is therefore likely that the ACP will develop an interface card that will allow VME devices to be plugged transparently into the Branchbus Switch.

With very high performance RISC processors in the Branchbu Switch Crate, the experimenter's ACP system seems likely to merge with that designed for theorists. However, the processors may continue to differ because large theoretical problems, with their very regular accesses to memory, tend to have a very high cache miss rate in conventional cache designs. Multi level cache may be the answer to this problem in an approach that could be equally effective for both classes of HEP problems. (In fact this may be the way in which manufacturers solve the static RAM speed requirements mentioned earlier in an affordable way.) An alternative approach, which adds significant software and hardware complexity, could be called "anticipatory" cache. In regular theoretical problems the data that will be required from memory is known well ahead of time. Means could be devised to have programs inform the hardware of anticipated memory fetches so the data could be moved from slow memory to cache before it is needed.

The way in which the huge anticipated increases in computing are going to be brought into on line systems for data acquisition and triggering for the high rate SSC era detectors is very likely going to be the subject of ACP development work over the next few years. There is also underway a project to develop particularly efficient work station tools for doing analysis. Clearly when the new processor systems reduce the turn around time to pass through a DST data base from days to less than an hour, it will be inappropriate to continue spending days lining up calls to HBOOK (a histogramming package) for a next analysis run. Macintosh like human interfaces, adjusted to physicist needs and abilities, will be used on Apple or, perhaps, Sun or other workstations.

It is clear that the opportunity exists to continue development of usable extremely high performance, yet affordable, parallel machines for high energy physics and other sciences hungry for computer power. This opportunity is really an obligation given the strongly felt need.

5. REFERENCES

a. Permanent address: Centro Brasileiro de Pesquisas Físicas; Rua Xavier Sigaud 150; Rio de Janeiro, RJ, 22290 Brazil.

b. Permanent address: University of Arizona, Tucson, AZ 85721

1. Gaines, Irwin and Nash, Thomas, Use of New Computer Technologies in Elementary Particle Physics, *Ann. Rev. Nucl. Part. Sci.* 37, 177-212 (1987)
2. Fox, G.C., Otto, S.W. *Phys. Today* 37(5): 50-59 (1984); Fox,G. The performance of the Caltech Hypercube in scientific calculations. In *Supercomputers - Algorithms, Architectures and Scientific Computation*, ed. F.A. Masten, T. Tajima, Univ. Texas (1985); Seitz, C.L., *Commun. ACM* 28 22 (1985).
3. Kunz, P. F., *Nucl. Instrum. Methods* 135 435-440 (1976); Kunz, P.F., Gravina, M., Oxoby, G., Trang, Q., Fucci, A., Jacobs, D., Martin, B., and Storr, K., The 3081/E Processor in *Proc. Three Day In-Depth Rev. on the Impact of Specialized Processors in Elementary Part. Phys.*, Padova, 1983, 83-100 (1983).
4. Gaines, I., Areti, H., Atac., R., Biel, J., Cook, A., Fischler, M., Hance, R., Husby, D., Nash, T., and Zmuda, T., The ACP Multiprocessor System at Fermilab, *Comp. Phys. Comm.* 45 323-329 (1987); Biel, J., Areti, H., Atac., R., Cook, A., Fischler, M., Gaines, I.,Hance, R., Husby, D., Nash, T., and Zmuda, T., Software for the ACP Multiprocessor System,*Comp. Phys. Comm.* 45 331-3379 (1987).
5. Hance, R., Areti, H., Atac, R., Biel, J., Cook, A., Fischler, M., Gaines, I., Husby, D., Nash., T., and Zmuda, T., The ACP Branchbus and Real Time Applications of the ACP Multiprocessor System, *IEEE Trans. Nucl. Sci*, NS-34, 878-883 (1987).
6. *Second Generation ACP Multiprocessor System: System Specification Document*; July 25, 1988 and revisions; Fermilab Advanced Computer Program internal document; unpublished. Availability of ACP publications may be determined by reference to the file at the HEPNET location, `fnacp::acpdocs_root:[docs]doclist.doc`
7. Gottlieb, S.A. , et al., *Phys. Rev. Lett.* 55 1958 (1985); Christ, N.H. and Terrano, A.E., *Phys. Rev. Lett.* 56 111 (1986).
8. Christ, N.H. and Terrano, A. E., *IEEE Trans. Comp.* C-33(4) 344 (1984); Beteem, J., Denneau, M. and Weingarten, D., *J. Stat. Phys.*, 43 1171 (1986); Albanese, M., et al., The APE Computer, ROM2F/87/005 (1987).
9. *CANOPY -- ACP MAPS Software - Specifications Document*, Fermilab Advanced Computer Program internal document; March 26, 1988 and revisions; unpublished.
10. Mackenzie, P., Eichten, E., Hockney, G., Thacker, H.B., Atac, R., Cook, A., Fischler, M., Gaines, I., Husby, D., Nash, T., ACPMAPS: The Fermilab Lattice Supercomputer Project, Proceedings International Symposium *Field Theory on the Lattice*, Seillac, France, Sept. 28- Oct. 2, 1987, Nucl. Phys.B (Proc. Suppl.)4, pp 580-584 (1988).