The Architecture of a Homogeneous Vector Supercomputer

John L. Gustafson, Stuart Hawkinson, and Ken Scott

Floating Point Systems, Inc.
Beaverton, Oregon 97005

Abstract

A new homogeneous computer architecture developed by FPS combines two fundamental techniques for high-speed computing: parallelism based on the binary *n*-cube interconnect, and pipelined vector arithmetic. The design makes extensive use of VLSI technology, resulting in a processing node that can be economically replicated. Processor *nodes* incorporate high-speed communications and control, vector-oriented floating-point arithmetic, and a novel dual-ported memory design. Each node is implemented on a single circuit board and can perform 64-bit floating-point arithmetic at a peak speed of 16 MFLOPS. Eight nodes are grouped together with a system node and disk support to form *modules*. These modules, housed in cabinet-sized packages, are capable of 128 MFLOPS peak performance and make up the smallest homogeneous units of larger systems. The new FPS system achieves a careful balance between high-speed communication and floating-point computation. This paper describes the new architecture in detail and explores some of the issues in developing effective software.

I. Introduction

The quest for increased computational power in scientific computing and the limits of physical electronic devices have led to the exploration of new architectures as alternatives to traditional monolithic designs [9, 2]. Multiprocessor designs hold the promise of tremendous performance increases, provided the interconnection network can support the parallelism inherent in the computation. Vector pipelines provide significant performance increments, exploiting finer-grained parallelism. Further advantage is gained by using parallel functional units to overlap address calculations with memory references, floating-point adds, and floating-point multiplies [1].

Large scientific applications are sometimes easily partitioned among processors using a shared memory, yet most are just as amenable to distributed memory designs [3, 4]. Shared memory systems are expensive when scaled to large dimensions because of the rapid growth of the interconnection network; the distance from memory to the processing elements also degrades performance by increasing latency [8]. Large system configurations are most readily realized with distributed memory based on a limited form of interconnection, such as the pyramid or the binary *n*-cube [5]. Memory latency can be greatly reduced when each processor has its own high-

speed store. Moreover, the cost of switching and the time to route messages is much smaller on such statically configured systems. With this view, much current computer architecture research has focused on the use of ensembles of identical processors in homogeneous configurations that employ message passing over limited forms of static interconnects [7, 8].

Floating Point Systems (FPS) has developed a homogeneous computer, the FPS T Series, based on the binary *n*-cube interconnection scheme. The individual nodes are 64-bit floating-point computers that combine vector arithmetic, dual-port memory, and fast communications links between nodes. The peak performance of these nodes is 16 MFLOPS. The FPS T Series is built from modules containing eight of these nodes connected to each other and to a system support ring. These modules, with an aggregate performance of 128 MFLOPS, may be combined to form even larger systems that promise orders of magnitude increases in computing speed per dollar over today's supercomputers.

II. Processor Node Architecture

An individual processor element is called a *node*. It contains a control processor, floating-point arithmetic, dual-port memory, and communication links to other nodes (see Figure 1). The FPS T Series design provides all of these functions on a single printed circuit board. Each of the major elements of the node has been implemented with advanced, cost-effective VLSI technology, in contrast with more traditional bit-slice designs.

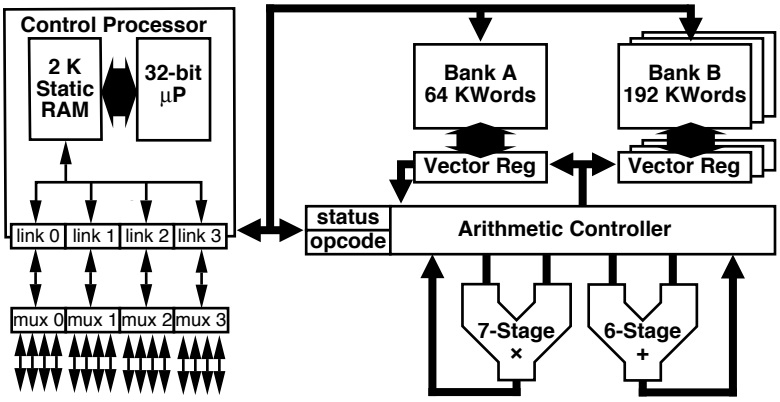

Figure 1. The FPS T Series processor

Control

The ability to interpret and execute programs resides in the central Control unit. The T Series control unit is a 32-bit CMOS microprocessor with the following functional features:

- 7.5 MIPS instruction rate
- Byte addressability (4 GByte address space)
- 2048 bytes of on-chip RAM (single processor cycle)
- 3-cycle minimum access time for off-chip memory
- Four bidirectional serial communications links
- Stack-oriented instruction set with variable operand sizes
- Two-level process priority and interrupt services

The control processor executes system and user applications code and it also serves to arrange vector operands to be sent to the vector arithmetic hardware. The control processor can execute integer arithmetic and gather/scatter operations in parallel with the vector unit, and it provides inter-node communications via the serial links.

All features of the microprocessor are directly accessed through a high-level language called Occam. Occam differs from languages like Pascal or C in that it directly provides for the execution of parallel, communicating processes. Channel commands can make direct data transfers between concurrent processes. A single process can be constructed from a collection by specifying sequential, alternative or parallel execution of the constituent processes. This combination of program structure and integrated communication allows Occam to describe the control and data flow for virtually any scientific computing algorithm, and to control the high-level operation of the vector arithmetic unit (see below).

Memory

An essential feature of a computer's architecture is its central memory, which supplies both instructions and operands to the processing units. The main memory of each FPS T Series node consists of 1 MByte of dual-ported dynamic RAM. The control processor and communications links read and write 32-bit words through a conventional random-access port, while the vector arithmetic unit makes use of a collection of vector registers closely coupled with main memory. A vector register can be loaded with an entire 1024-byte row of memory, in parallel (see Figure 1), in the same time that it would have taken to read or write a single 32-bit word. There is one parity bit for each byte in memory.

The control processor views the memory as a single bank of 256K words (32-bit). The vector arithmetic unit views memory as two banks of vectors, with 256 vectors in one bank and 768 vectors in the other, aligned on 1024-byte boundaries. Thus, for 32-bit operations, the vectors are 256 elements long, while for 64-bit operations, the vectors are 128 elements in length. The division of memory into two banks permits two inputs in parallel to the arithmetic unit on each cycle (125 ns). The output of the arithmetic unit shifts results into either or both banks. Hence, operations such as SAXPY, Vector Add, and Vector Multiply proceed at the full speed of the arithmetic components, without being limited by available memory bandwidth. This dual-bank memory organization allows the node to function without the need for auxiliary data registers or cache.

The control processor can access a 4-byte word in 400 ms. Its effective bandwidth to RAM is therefore

$$(4 \text{ bytes}) / (0.4\ \mu s) + 10 \text{ MB/s}$$

A primary use for the control processor is to gather operands into a contiguous vector, and scatter results back to random locations in memory. To move a 64-bit operand from one memory location to another requires two 32-bit reads and two 32-bit writes, which take a total of 1.6 μs. This is the gather-scatter time within a node. For 32-bit operands, it is 0.8 μs per element.

An entire row of data can be moved to or from a vector register in only 400 ns; this means that the effective bandwidth between memory and a vector register is

$$(1024 \text{ bytes}) / (0.4\ \mu s) = 2560 \text{ MB/s}.$$

An application might make use of this extraordinary speed by moving data physically, rather than keeping linked lists of pointers to vectors, as for example, in pivoting rows of a matrix or sorting records.

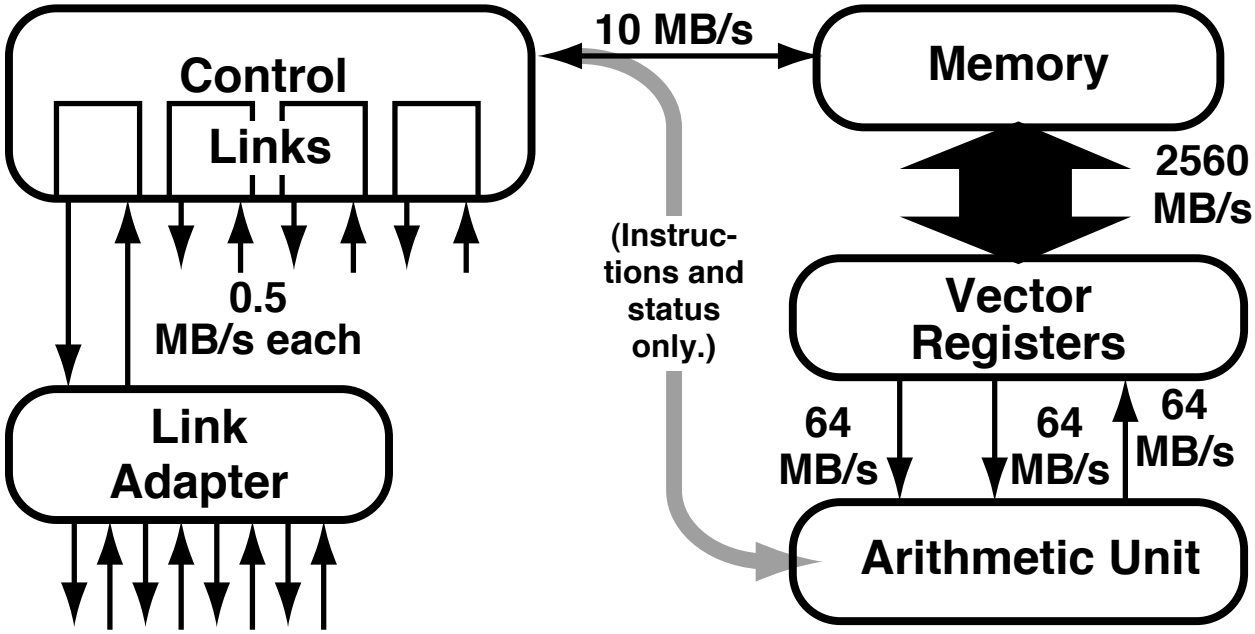

Figure 2. Processor bandwidths

The vector registers each supply data to the arithmetic unit at a maximum rate of one 32-bit word every 62.5 ns, or one 64-bit word every 125 ns. The vector register bandwidth supports two vector inputs and one vector output every 125 ns in 64-bit mode. Thus, its bandwidth is

$$(3 \text{ words}) \times (8 \text{ bytes/word}) / (0.125\ \mu s) = 192 \text{ MB/s}.$$

Arithmetic

The ability to perform high-speed arithmetic is essential in scientific computing. The arithmetic hardware in the FPS T Series consists of a floating-point adder, floating-point multiplier, interconnection hardware, and some sequencing hardware. The adder and multiplier each can produce a 32- or 64-bit result every 125 ns, yielding peak performance of 16 MFLOPS per node. Floating-point operations are performed using the proposed IEEE Floating-point standard format; however, gradual underflow is not supported. In 64-bit mode, the mantissa has approximately 15 decimal digits of precision (53 bits) and a dynamic range of roughly 10^{-308} to 10^{+308} (11-bit binary exponent).

The arithmetic units operate in pipelined mode. The adder has a six-stage pipeline. It can perform floating-point addition and subtraction in 32- and 64-bit modes, comparisons, and data conversions. The multiplier is five-stage in 32-bit mode and seven-stage in 64-bit mode. These pipeline lengths are appropriate for the vector access described above. Scalar operations can be efficiently performed by grouping like operations for level-order evaluation.

The arithmetic functional units are supervised by a preprogrammed micro-sequencer that implements a collection of vector arithmetic operations referred to as *vector forms*. The programmer only needs to describe the input and output vectors and the vector form desired. This frees the control processor for other tasks while vector operations are being executed. Scalars can be held in the input registers on each floating-point functional unit, and outputs from the functional units can be fed directly back as inputs to perform operations such as dot products and sums. This provides a wide range of useful vector forms without memory reference limitations. The complete arithmetic unit operates in parallel with the node control processor. The arithmetic unit only interrupts the controller when a vector operation has completed, or an error has occurred.

Communications

In a distributed computer system, communications channels are required for passing data between processors participating in a common computational process. The control processors of the FPS T Series contains drivers for four serial, bidirectional communications links each with a nominal rate of 0.9375 Mbytes/s. Every 8-bit byte is sent with two synchronization bits and one stop bit, and requires two acknowledge bits from the receiver. This results in a maximum unidirectional bandwidth of over 0.5 MB/s per link. The total bandwidth of the four links is thus over 4 MB/s. With all links operating, the control processor performance is degraded only slightly. The links operate via DMA transfers with a startup time of about 5 μs.

Each link is multiplexed four ways to provide a total of 16 bidirectional sublinks per node. With software support, these sublinks divide the available bandwidth. Two sublinks are used for

system communication, and two will often be utilized for mass storage or external I/O. This will typically leave 12 sublinks available for connection to other compute nodes.

A convenient way to interpret the relative bandwidths is with respect to the arithmetic processing time for 64-bit operations:

(Arithmetic Time) :	(Gather Time) :	(Link Transfer Time)
.125 μs	1.6 μs	16 μs

that are in the approximate ratios

$$1 : 13 : 130$$

Thus, a vector should enter into about 13 operations while gathering the next vector into an aligned, contiguous order. With this provision, the control processor can completely overlap the gather time with vector arithmetic, and the node can approach peak speed. Of course, if vectors are always aligned and elements contiguous, no such restriction applies. Similarly, roughly 130 operations should result from every 64-bit word that must be moved between nodes over a link.

III. System Description

The members of the T Series consist of a number of node processors connected as a binary n-cube. There are 2^n processors, with n connections per node. If we number the processors from 0 to 2^{n-1}, each processor is directly connected to all others whose numbers differ in only one binary digit. The binary n-cube can be mapped onto many important applications topologies, including meshes (up to dimension n), rings, cylinders, toroids, and even FFT butterfly connections of radix 2 [5,6]. Since the maximum number of connections between any two processors is n, long-range communication costs grow only as $O(\log_2 n)$.

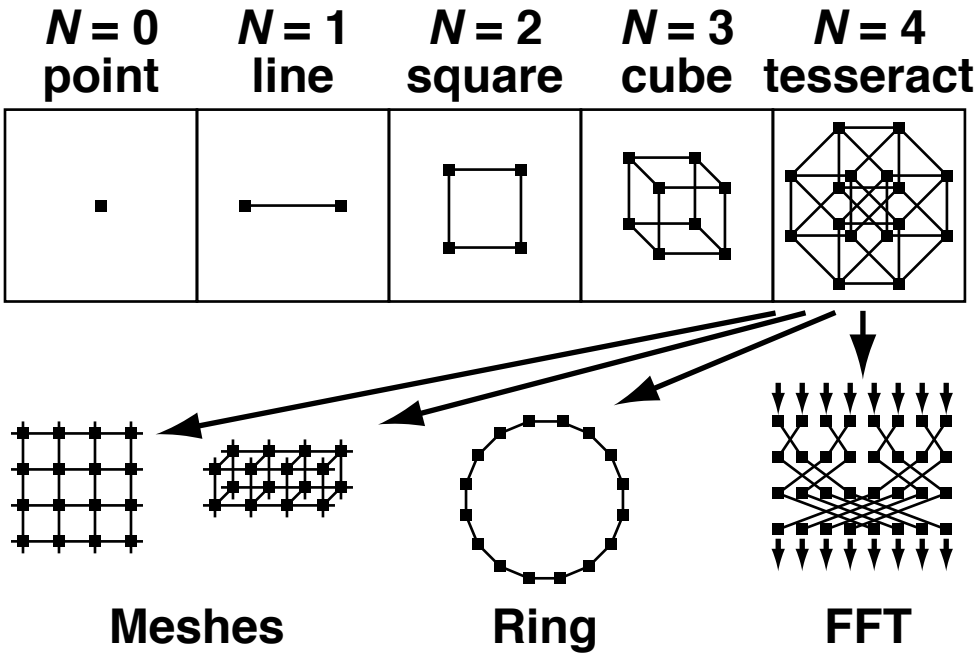

Figure 3. Binary n*-Cube Mappings*

Modules and System Ring

A processor node is constructed on a single etched circuit board. Eight nodes are combined with disk storage and a system board to form a module. Such a module has 128 MFLOPS peak floating-point performance, and 8 MB of user RAM. The local inter-node communications bandwidth is over 12 MB/s, while the system board can support 0.5 MB/s to an external connection.

The system board provides input/output and management functions. It is connected to the nodes by a thread of communications links that traverses the eight processor nodes. The system boards are directly connected by communications links to form a system ring that is independent of the binary *n*-cube network (connecting the processor nodes). The primary function of the system disk is to record memory "snapshots" which checkpoint computations for error recovery, and to backup snapshots from other modules. The user is able to specify the interval between snapshots. About 10 minutes provides a good compromise between time spent to record memory and interval between restart points. It takes about 15 seconds to take a snapshot, regardless of configuration.

The module requires three links for intramodule hypercube network communications, while the system board connections require two links from each processor node. This reduces remaining links to 11 for hypercube and external communications.

Two modules (16 nodes) form a *cabinet*, or 4-cube (a tesseract). A cabinet is modular and self-contained in a standard 19-inch rack-mounted assembly. With power supplies, system disks, and air-cooling fans, it does not require any special "computer room" facilities. Larger systems are

simply assembled from these units by interconnecting cables. Connections up to 40 feet can be made without special consideration.

Larger Configurations

A four-cabinet (64-node) system has an aggregate peak speed of 1 GFLOPS and total user memory of 64 Mbytes. Eight system-disk units provide backup and restart capability. This configuration of the T Series can be located in many laboratory and computing facilities. The air-cooled unit requires no special facilities beyond normal air conditioning, and the power requirements are supplied by typical 220 VAC services.

There are enough links per node to permit a 14-cube to be constructed as the largest T Series configuration. Using two links per node for external I/O and mass storage systems, a maximum-sized 12-cube consists of 4096 nodes arranged as 256 cabinets (4-cubes). Such a system has over 65 GFLOPS peak processing performance and 4 Gbytes of primary RAM storage. Special facilities would be required to house the largest configurations.

Because the system is homogeneous, i.e., each module is identical and contains identical connections to other modules, programming is greatly simplified. This homogeneity also insures that the balance between computing speed, main storage, mass storage, and external I/O can be preserved as configurations become large. The specifications of any sized FPS T Series can be derived from the properties of the individual modules.

IV. Conclusion

The incorporation of high-speed vector processing into a homogeneous parallel architecture has resulted in a scientific computer with performance scalable over three orders of magnitude. The use of a dual-ported dynamic RAM achieves a new level of processor integration that eliminates the need for separate data registers or cache. Parallel floating-point adders and multipliers, accessed by standard vector operations, provide a close match to scientific computing algorithms.

With nodes organized into eight-processor modules, each with system disk and I/O services, the new architecture can be viewed as a truly homogeneous system. The FPS T Series, incorporating VLSI components to make the system both cost-effective and compact, provides a careful balance between processor speed, memory access, and interprocessor communications bandwidth.

References

[1] Charlesworth, A. E., "An Approach to Scientific Array Processing: The Architectural Design of the AP-120B/FPS-164 Family." *IEEE Computer*, Volume 14, Number 9 (1981), 18–27.

[2] Charlesworth, A. E., and Gustafson, J. L., "Introducing Replicated VLSI to Supercomputing: The FPS-164/MAX Scientific Computer". *IEEE Computer*, 19, 3 (1986).

[3] Fox, G. C., "Decomposition of Scientific Problems for Concurrent Processors". *Technical Report CALT-68-986*, Computer Science Dept., Caltech, 1983.

[4] Fox, G. C., and Otto, S. W., "Algorithms for Concurrent Processors". *Physics Today*, 37, 5 (1984), 50–59.

[5] Pease, M. C., "The Indirect Binary *n*-Cube Microprocessor Array". *IEEE Transactions on Computers*, C-26, 5 (1977), 458-473.

[6] Saad, Y., and Schultz, M. H., "Topological Properties of Hypercubes," *Technical Report YALEU/DCS/RR-389*, Computer Science Dept., Yale Univ., 1985.

[7] Seitz, C. L., "Experiments with VLSI Ensemble Machines". *Journal of VLSI Computing Systems*, 1 (1984).

[8] Seitz, C. L., "The Cosmic Cube," *Communications of the ACM*, 28, 1 (1985), 22–33.

[9] Seitz, C. L., and Matisoo, J., "Engineering Limits on Computer Performance". *Physics Today*, 37, 5 (1984), 38–45.