PNC TN9410 93-148

THE CONCEPT OF PARALLEL INPUT/OUTPUT PROCESSING FOR AN ELECTRON LINAC

June, 1 9 9 3

OARAI ENGINEERING CENTER

POWER REACTOR AND NUCLEAR FUEL DEVELOPMENT CORPORATION

PNC TN9410 93-148
June, 1993

The Concept of Parallel Input/Output Processing for an Electron Linac

T. Emoto

Abstract

The instrumentation of and the control system for the PNC 10MeV CW electron linac are described. A new concept of parallel input/output processing for the linac has been introduced. It is based on a substantial number of input/output processors(IOP) using beam control and diagnostics. The flexibility and simplicity of hardware/software are significant advantages with this scheme.

PNC TN9410 93-148
1993 年 6 月

電子線形加速器に関する並列入出力処理計算機の概念

江本 隆

要旨

動力炉・核燃料開発事業団の10 MeV CW電子線形加速器に関する計測系及び制御系について述べる。電子線形加速器に関する新しい概念である並列入出力処理計算機について紹介する。それは数多くの入出力処理計算機(Input Output Processor)を用いて電子ビームの制御と診断を行うものである。このような手法を用いることにより、ハードウエア/ソフトウエアの簡略化と柔軟性が大きな利点となる。

Contents

List of Figures

1. Introduction

Recently much progress has been made in one of the methods called parallel architecture.[1] Today parallel architecture, which uses RISC (Reduced Instruction Set Computer) processors, advance in processor-to-processor communication schemes, solid support from mainstream vendors, and parallel processing, can be found not only in academic or large supercomputing facilities but also in the main stream of commercial computing.[2]

However, such a parallel processing is constructed mainly for computation-intensive applications such as supercomputers which tackle with complex scientific problems. The benefits of parallel architecture are also applied to control a large number of peripheral devices in a short period of time and new approach to control. This new concept is called *parallel input/output processor (PIOP)* in this paper. In general, a conventional mainframe computer input/output control scheme has been used to reduce central processing unit(CPU) overhead with dedicated processors for each peripherals, which are called IOP (input/output processor) shown in Figure. 1. Even mini/micro computers have this configuration by downsizing of computer hardware. A conventional mini/micro computer system is shown in Figure. 2, which has interface circuits instead of IOPs. The IOP scheme seems to be parallel processor architecture but input/output data and memory can not be shared and processed simultaneously because of a shared-bus system. In other words, PIOP consists of a large number of IOPs which have unique high-speed communication bus between each IOP, considerable fast computation power, and dedicate memory.

This paper reviews the discussion of the parallel processing and introduces a new concept of a prospective PIOP control system for a PNC electron linac facility.

2. Parallel processing

All parallel architecture use multiple processors. The differences among types of parallel computers lie in other areas. Parallel computers are classified by two qualifiers: The relationship of the memory to the processors and the number of instruction streams available to the system.

There are two general relationships between memory and processors: *distributed memory* and *shared memory*. In distributed memory systems, each processor has its own private memory. Shared memory systems have a single pool of memory to which all processors have access.

The other terms that differentiate parallel computers are *MIMD*(Multiple Instruction, Multiple Data) and *SIMD*(Single Instruction, Multiple Data). MIMD systems allow processors to work on separate instruction streams, or tasks, at the same time, while the processors in SIMD systems all operate on a single instruction stream simultaneously. Synchronous behavior is automatic in SIMD machine; MIMD systems require that synchronization be programmed in. A MIMD machine can simulate an SIMD machine, but not vice-versa. MIMD machines can make use of either shared or distributed memory, but only distributed memory is used in SIMD machines because multiple processor could not be used efficiently with a single instruction stream and a single memory pool.

The focus of this paper is on distributed memory, MIMD systems. Systems of this kind are composed of a set of *nodes*(or computing elements), each of which comprises a main processor, memory, and interface to the network. Each node has the computing power of a stand-alone workstation. Nodes process information independent of one another and communicate by sending and receiving massages. This independence gives these systems what is called a *loosely-coupled* architecture. In another word, the term loosely coupled describes a set of processor running relatively independently of each other on a network.

A loosely-coupled system design facilitates this flexibility.[3] It is possible to have a system with specialized node for numeric, symbolic, or input/output processing, all integrated in one system. These nodes can be programmed either to cooperate to solve a single problem or to work on several different problems at once.

In a distributed memory MIMD system, every computing node is a complete computer with its own local memory. Because other nodes can not see into a node's private memory space, results must be passed between nodes over a communication network. The performance of MIMD is as much affected by the speed of its communication network as it is by the speed of the processing elements. Perhaps, it would be better to say that the balance between communication-time and computation-time in a message-passing machine varies from problem, according to the algorithm used and the topology and performance of the communications network utilized.

There are as many ways to connect distributed processors as there are ways to use a constructor set, and most of them have been tried by someone, somewhere (Figure. 3). Pipeline, arrays, meshes, cubes, hypercubes, trees, rings, and toruses all have their advocates.

The PIOP architecture have selected the hypercube configuration because it offers the fastest and most reliable communications network. The hypercube network is organized so that the connections between computing elements form cubes. As more computing elements are added in a system configuration, the cube grows to larger dimensions (Figure. 3).

The *size* of a hypercube is also called its *order*. Both size and order refer to the dimension of the cube (a three-dimensional hypercube is an order three hypercube). A two-dimensional hypercube is a square. The number of computing elements in any size of hypercue is written as 2^x, where x is the order, and (2^6) has 64 computing elements. The hypercube network provides multi-step communication paths by connecting each computing element to a subset of the other computing elements. A key advantage of the hypercube is its ability to connect thousands of individual computing elements. One measure of a network's efficiency is the number of steps required to move data between the farthest computing elements in a system. In the highly efficient hypercube topology, the greatest distance (or the number of steps) between computing elements is the same as the cube's dimension. An order six hypercube containing 64 computing elements, for example the maximum number of steps a message must travel between any computing elements is only six.

3. System configuration

3.1 Outline of the system configuration

The distributed-memory MIMD or SIMD systems and loosely-coupled hypercube architecture are chosen to the configuration of PIOP architecture as previous discussions. The system diagram (Figure 4) shows that the PIOP system operates in association with a conventional computer, which is called the front-end computer (or host), and a few ten of processors which are called nodes. The front-end computer and each node are connected with two different communication lines. One is a conventional VME bus-based system that all processors must shear the same bus to access to the other processors one by one at a time. However, the VME bus is an industrial standard bus system, which has the advantages of many variety of hardware and software products. The other communication line is a high speed communication line (massage-passing network, loosely-coupled hypercube communication line) that supports message traffic between nodes. Messages can convey information between nodes, and also synchronize node activities. Messages containing data or code are copies of information resident on the sending node. They are sent through the message-passing network and delivered to other nodes. In general, the time required to pass messages is dependent on the speed of the hardware and the way the network is interconnected. A balanced system will exhibit message-passing performance that is comparable to computational performance.

3.2 PIOP nodes

Each PIOP node (Figure. 5) consists of a processor, distributed memory, high-speed communication ports, VME bus interface, and peripheral interface such as A/D and D/A converters. A digital signal processor(DSP) is a candidate for PIOP node processor because any application for analog electronic circuitry can be duplicated using DSP with the flexibility and simplicity of the analogy part of hardware. For parallel processing system, the processor should have both a floating-point CPU and parallel inter-processing communication channel ports which allow direct processor to processor communication over a point to point. That high speed communication ports are important components of the PIOP node to achieve the loosely coupled hypercube communication lines.

3.3 programming consideration

A processing in a loosely-coupled system is a sequential part of a complete parallel application. This sequential part runs on a single system node. To make most efficient use of a loosely-coupled parallel computer, a set of processors that are executed simultaneously (in parallel). Processors perform different tasks and exchange information as required to achieve the common goals of the application. The processors in a loosely-coupled parallel system operate independently of one another. As a result, processes on different nodes are executed asynchronously. There is no timing relationship between them. Communication serves both to synchronize presses and to exchange code and data information among processes.

The message-passing calls include synchronous or blocking sends and receives. This is probably the simplest to visualize. When a process issues a request to receive a message, the process executes no further instructions until a message arrived at the node at a time the request is made, the process will do nothing until message arrives. If the message had arrived at the node before the process made the request, the process would wait only until the message is copied into the process memory from the system buffer in which it was temporarily stored. Synchronous message passing calls are essential for correct implementation of many algorithms, and are also useful for initial development and debugging of applications that will ultimately use asynchronous message passing.

It is more efficient to handle message asynchronously. Asynchronous receive allow processes to alert the operating system that certain messages are expected and should delivered to the process as soon as they arrive, even though the process may be busy doing other work at the time. The process does not idly wait for the massage to arrive, it continues to execute instructions until the information in the message is required. A process can also send message asynchronously same scheme.

Another way of handling message in a timely fashion is called *interrupt handling*. Like asynchronous message handling, the process continues while waiting for the message to arrive. The difference is that with interrupt handling, the process receives an interrupt as soon as the message arrives so it can be processed immediately, rather than having the receiving process post the receive when it is ready to receive it. Afterwards, the process can return to what it was doing when the interrupt occurred. These three different massage passing; asynchronous, synchronous, and interrupt are applied to the PIOP system software development not only process efficiency and also the stage of system development.

4. Discussion

The PIOP system will be applied to a beam control for an electron linac. The beam control contains controls and status displays for each individual linac equipment (an electron gun, an injection system, a beam transport system, and a beam dump, etc). The beam control is classified as part of a central control system which can be divided into five parts; the beam control, a operator's console, utility equipment for water-cooling, radiation monitor systems, and gas processing equipment. Figure. 6 illustrates their relation. These systems are connected with the standard communication network(Ethernet with the TCP/IP protocol). The communication network is supervised by the system control processors which also manage the whole electron linac operation. Those processors are responsible for not crucial operations from the response time point of view because the processors manage the operator's console and utility equipment.

The PIOP system (served as the beam control system) is interfaced with the linac equipment and each PIOP node connected with VME-bus and the high speed communication lines as described chapter 3.2. Each PIOP node is connected with high speed communication lines as horizontally (or hypercube shape), which makes each node communicate with another node a short time as compared with shared bus system. A combination with high speed communication lines and the DSP (as node processor) makes fast data processing system for large number input parameters simultaneously. That data processing could not be achieved in conventional system. Such examples as beam current signals from each accelerator tube section are digitized by DSP and A/D converter and processed in each node processor at the same time.

5. Conclusion

The new concept of PIOP for the electron linac has been introduced. A substantial number of input/output processors are connected with high speeed communication networks as hypercube shape. The PIOP system is used to provide function which could not be obtained through the conventional data processing system.

The designing of PIOP system hardware and software for the electron linac is in progress.

References

[1] B. M. Boghosian, **COMPUTER IN PHYSICS** JAN/FEB **4-1** (1990) 14.

[2] D. Pountain and J. Bryan, **BYTE** August **17-8** (1992) 112.

[3] F. Rinaldo and S. Wolbers, **COMPUTER IN PHYSICS** MAR/APR **7-2** (1993) 184.

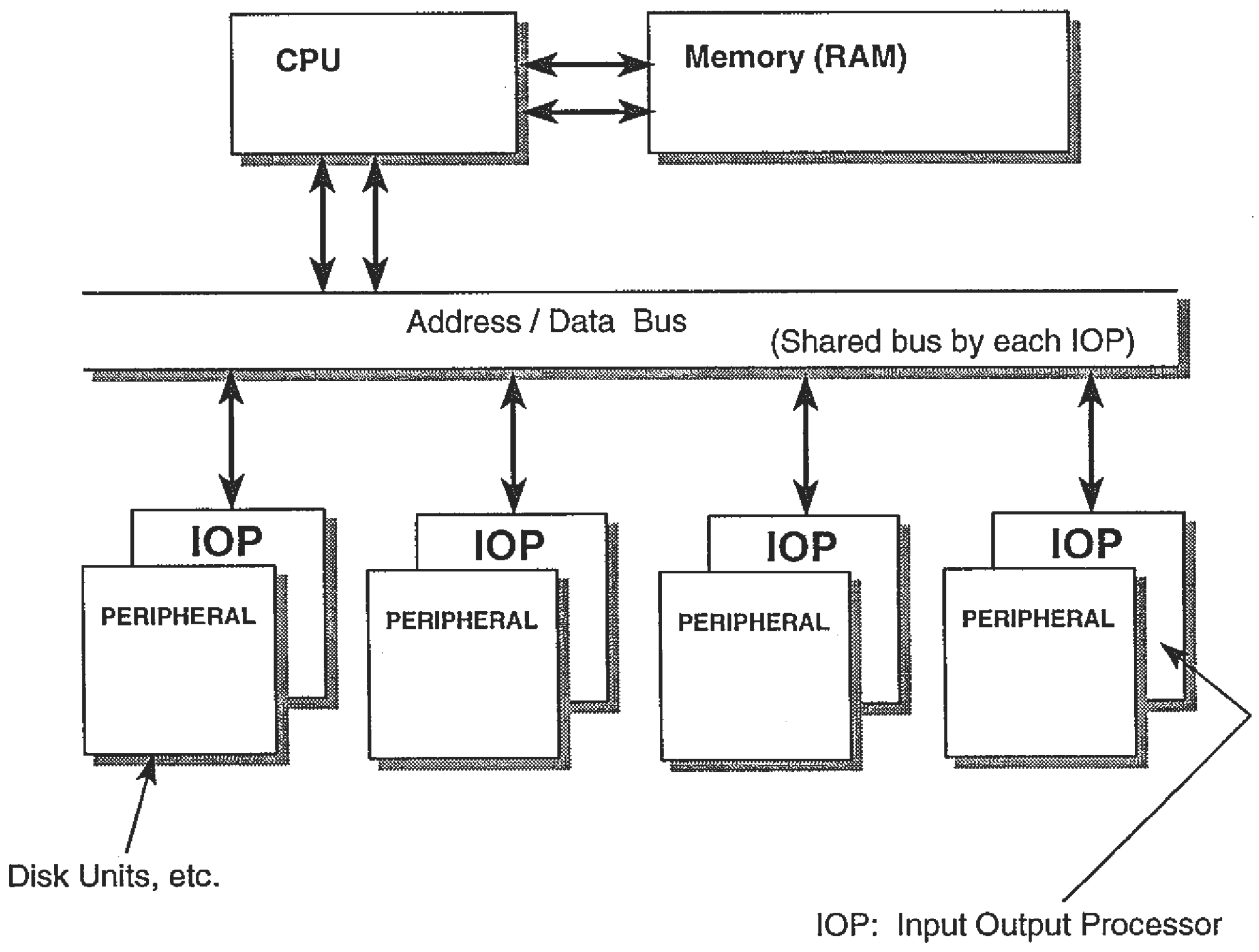


Figure. 1 A conventional mainframe computer system.

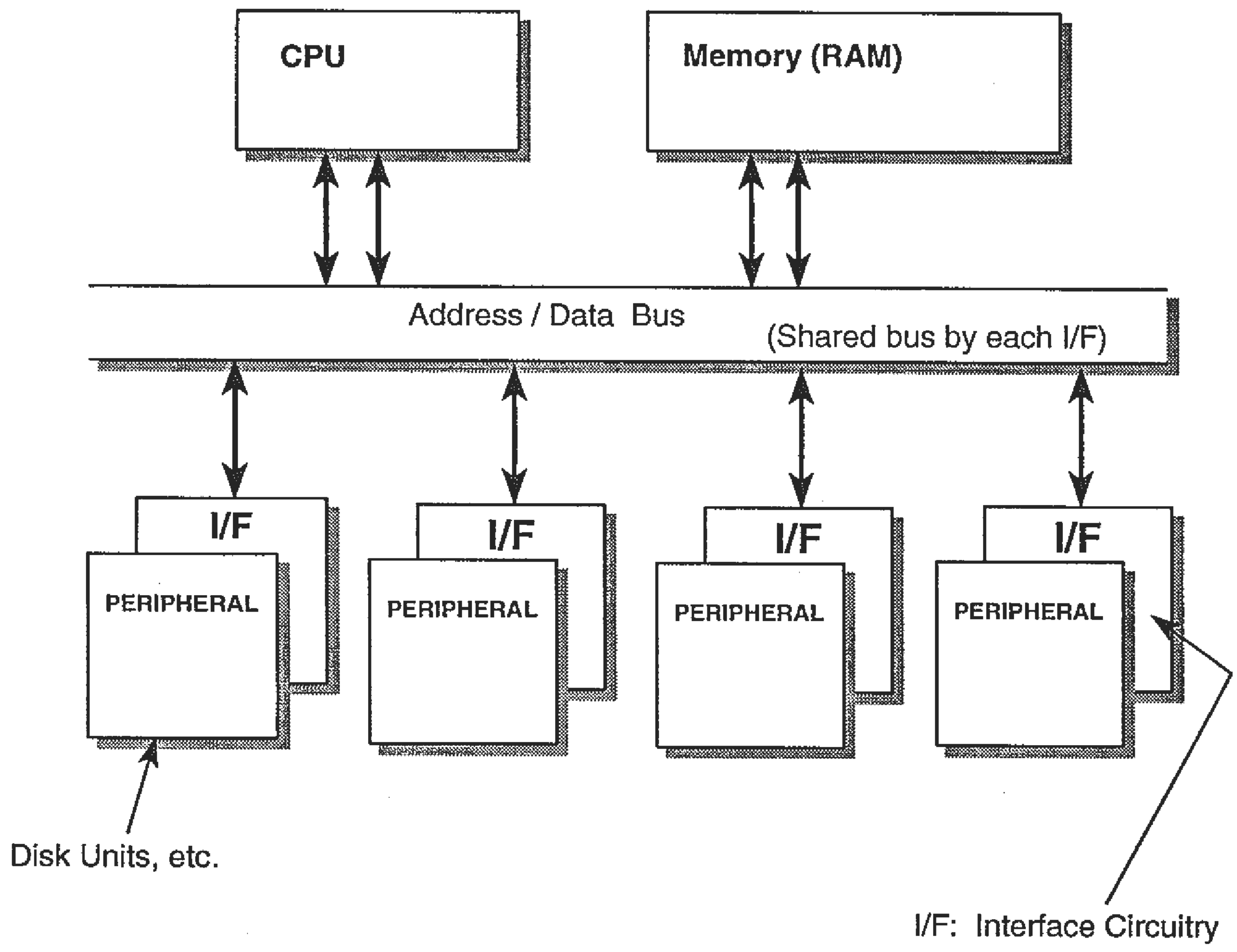
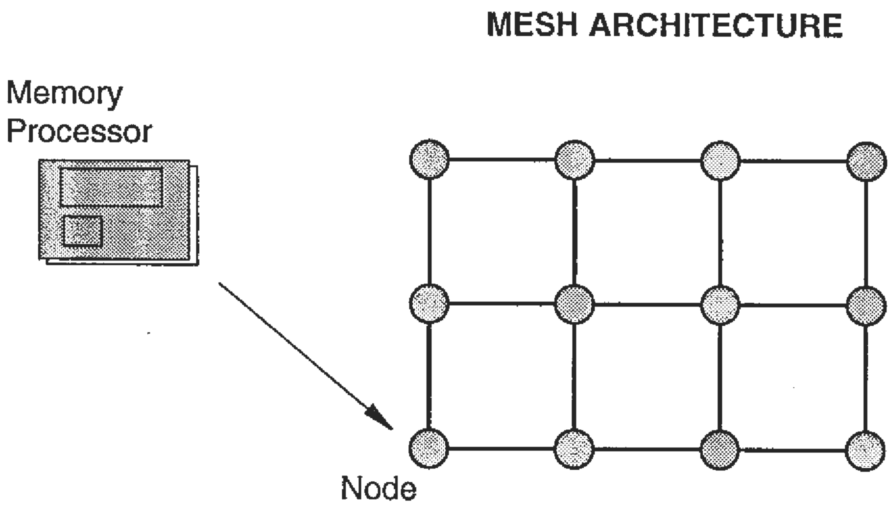


Figure. 2 A conventional mini/micro computer system.



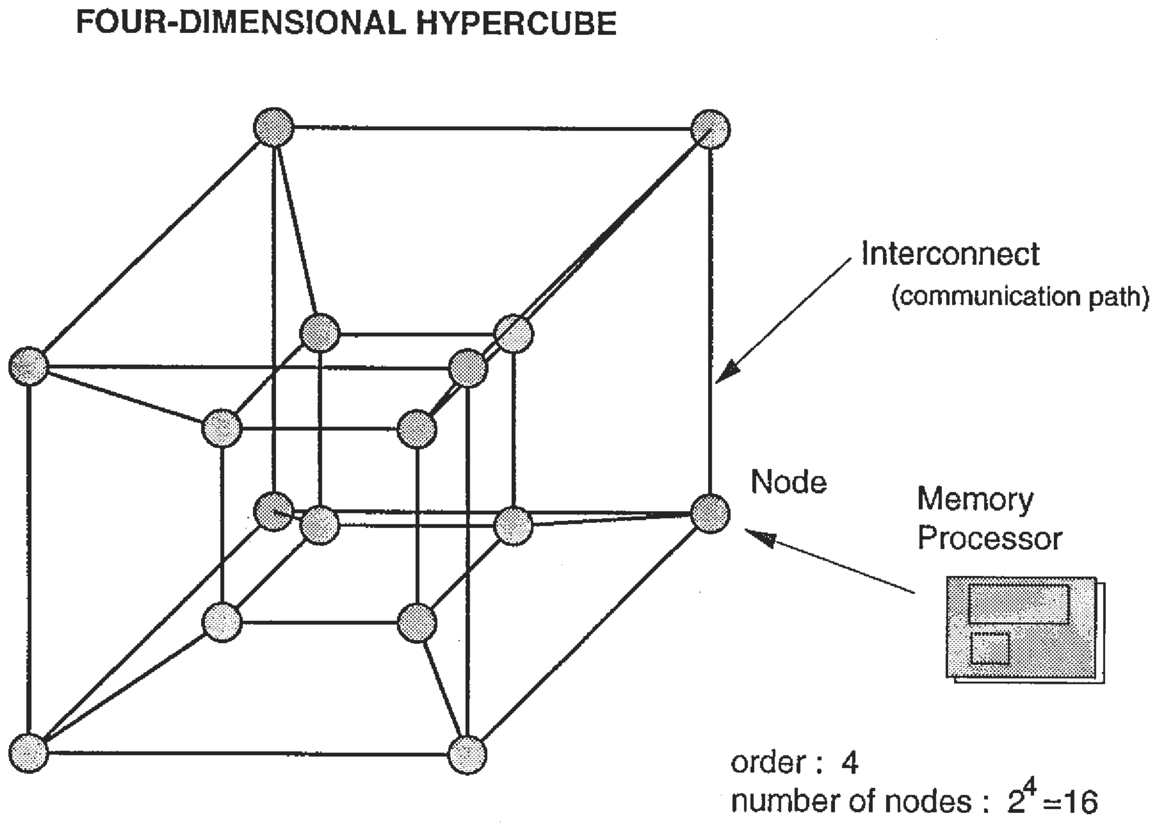


Figure. 3 Multi-processor topologis

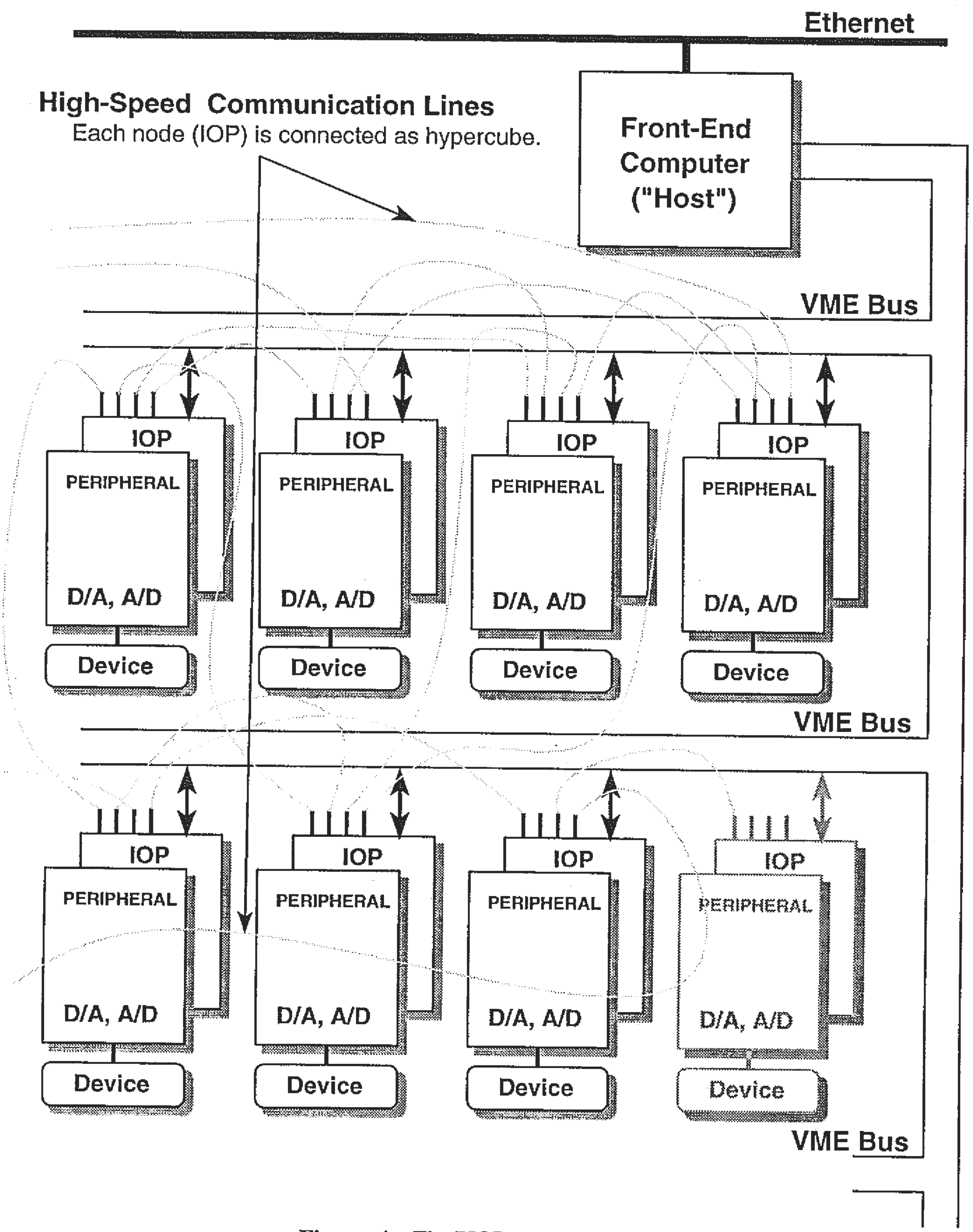

Ethernet
High-Speed Communication Lines
Each node (IOP) is connected as hypercube.
Front-End Computer ("Host")
VME Bus
IOP
PERIPHERAL
D/A, A/D
Device
VME Bus
VME Bus

Figure. 4 The PIOP system diagram.

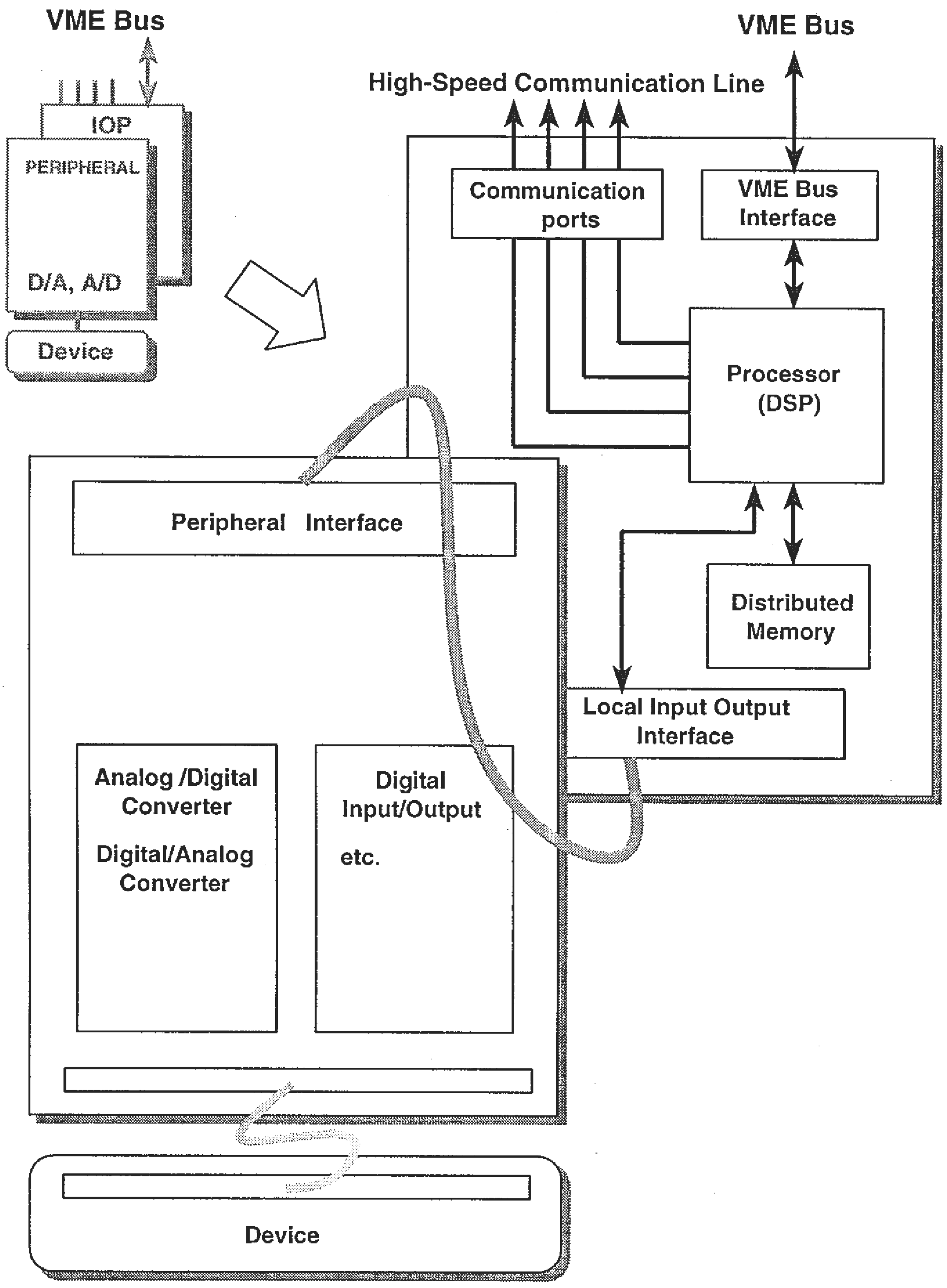


Figure. 5 The PIOP node diagram.

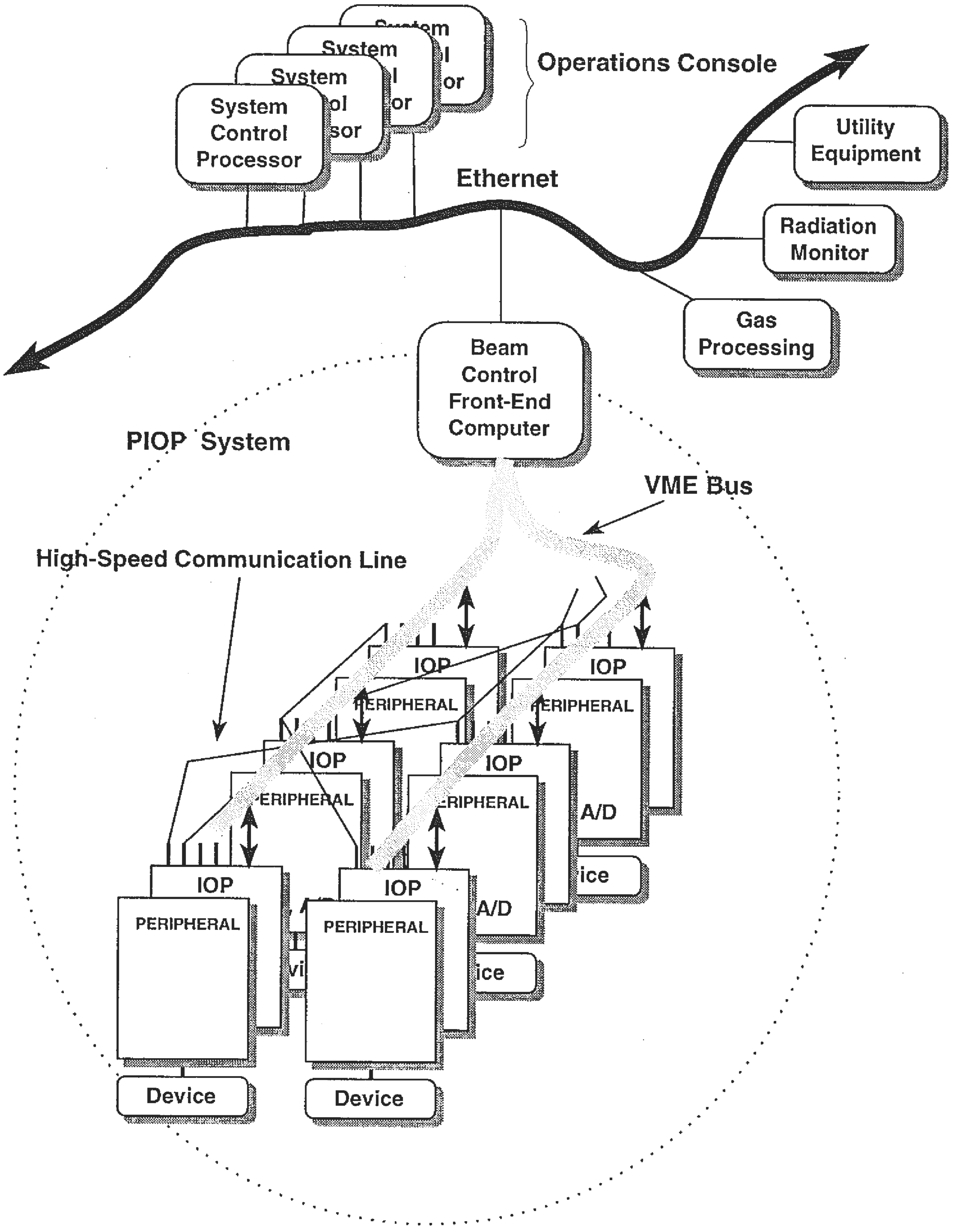


Figure. 6 The linac control system.