Chapter 1

OVERVIEW OF HIGH PERFORMANCE COMPUTERS

Aad J. van der Steen
Dept. of Computational Physics
Utrecht University
3508 TD Utrecht
The Netherlands
steen@phys.uu.nl

Jack Dongarra
Computer Science Department
University of Tennessee
and
Oak Ridge National Laboratory
Oak Ridge, Tennessee, USA
dongarra@cs.utk.edu

Abstract The overview given here concentrates on the computational capabilities of the systems discussed. To do full justice to all assets of present days high-performance computers one should list their I/O performance and their connectivity possibilities as well. However, the possible permutations of configurations even for one model of a certain system often are so large that they would multiply the volume of this report, which we tried to limit for greater clarity. So, not all features of the systems discussed will be present. Still we think (and certainly hope) that the impressions obtained from the entries of the individual machines may be useful to many. We also omitted some systems that may be characterized as "high-performance" in the fields of database management, real-time computing, or visualization. Therefore, as we try to give an overview for the area of general scientific and technical computing, systems that are primarily meant for database retrieval like the AT&T GIS systems or concentrate exclusively on the real-time user community, like Concurrent Computing Systems, are not discussed in this report. Al-

though most terms will be familiar to many readers, we still think it is worthwhile to give some of the definitions in section 2 because some authors tend to give them a meaning that may slightly differ from the idea the reader already has acquired.

1. Introduction

Before going on to the descriptions of the machines themselves, it is important to consider some mechanisms that are or have been used to increase the performance. The hardware structure or *architecture* determines to a large extent what the possibilities and impossibilities are in speeding up a computer system beyond the performance of a single CPU. Another important factor that is considered in combination with the hardware is the capability of compilers to generate efficient code to be executed on the given hardware platform. In many cases it is hard to distinguish between hardware and software influences and one has to be careful in the interpretation of results when ascribing certain effects to hardware or software peculiarities or both. In this chapter we will give most emphasis to the hardware architecture. For a description of machines that can be considered to be classified as "high-performance" one is referred to (Culler et al. 1998, van der Steen 1995).

2. The main architectural classes

Since many years the taxonomy of Flynn (1972) has proven to be useful for the classification of high-performance computers. This classification is based on the way of manipulating of instruction and data streams and comprises four main architectural classes. We will first briefly sketch these classes and afterwards fill in some details when each of the classes is described.

- **SISD** machines: These are the conventional systems that contain one CPU and hence can accommodate one instruction stream that is executed serially. Nowadays many large mainframes may have more than one CPU but each of these execute instruction streams that are unrelated. Therefore, such systems still should be regarded as (a couple of) SISD machines acting on different data spaces. Examples of SISD machines are for instance most workstations like those of DEC, Hewlett-Packard, and Sun Microsystems. The definition of SISD machines is given here for completeness' sake. We will not discuss this type of machines in this report.

- **SIMD** machines: Such systems often have a large number of processing units, ranging from 1,024 to 16,384 that all may execute the same instruction on different data in lock-step. So, a single instruction manipulates many data items in parallel. Examples of SIMD machines in this class are the CPP DAP Gamma II and the Alenia Quadrics.

Another subclass of the SIMD systems are the vector-processors. Vector-processors act on arrays of similar data rather than on single data items using specially structured CPUs. When data can be manipulated by these vector units, results can be delivered with a rate of one, two and — in special cases — of three per clock cycle (a clock cycle being defined as the basic internal unit of time for the system). So, vector processors execute on their data in an almost parallel way but only when executing in vector mode. In this case they are several times faster than when executing in conventional scalar mode. For practical purposes vector-processors are therefore mostly regarded as SIMD machines. An example of such systems is for instance the Hitachi S3600.

- **MISD** machines: Theoretically in these type of machines multiple instructions should act on a single stream of data. As yet no practical machine in this class has been constructed nor are such systems easily to conceive. We will disregard them in the following discussions.

- **MIMD** machines: These machines execute several instruction streams in parallel on different data. The difference with the multi-processor SISD machines mentioned above lies in the fact that the instructions and data are related because they represent different parts of the same task to be executed. So, MIMD systems may run many sub-tasks in parallel in order to shorten the time-to-solution for the main task to be executed. There is a large variety of MIMD systems and especially in this class the Flynn taxonomy proves to be not fully adequate for the classification of systems. Systems that behave very differently like a four-processor NEC SX-5 vector system and a thousand processor SGI/Cray T3E fall both in this class. In the following we will make another important distinction between classes of systems and treat them accordingly.

- **Shared-memory systems**: Shared-memory systems have multiple CPUs all of which share the same address space. This means that the knowledge of where data is stored is of no concern to the user as there is only one memory accessed by all CPUs on an equal basis. Shared memory systems can be both SIMD or MIMD. Single-CPU vector processors can be regarded as an example of the former, while the multi-CPU models of these machines are examples of the latter. We will sometimes use the abbreviations SM-SIMD and SM-MIMD for the two subclasses.

- **Distributed-memory systems**: In this case each CPU has its own associated memory. The CPUs are connected by some network and may exchange data between their respective memories when required. In contrast to shared-memory machines the user must be aware of the location of the data in the local memories and will have to move or distribute these data explicitly when needed. Again, distributed-memory systems may be either SIMD or MIMD. The first class of SIMD systems mentioned which operate in lock step, all have distributed memories associated to the processors. As we will see, distributed-memory MIMD systems exhibit a large variety in the topology of their connecting network. The details of this topology are largely hidden from the user which is quite helpful with respect to portability of applications. For the distributed-memory systems we will sometimes use DM-SIMD and DM-MIMD to indicate the two subclasses.

As already alluded to, although the difference between shared and distributed-memory machines seems clear cut, this is not always entirely the case from the user's point of view. For instance, the late Kendall Square Research systems employed the idea of "virtual shared-memory" on a hardware level. Virtual shared-memory can also be simulated at the programming level: A specification of High Performance Fortran (HPF) was published in 1993 (Forum 1993) which by means of compiler directives distributes the data over the available processors. Therefore, the system on which HPF is implemented in this case looks like a shared-memory machine to the user. Other vendors of Massively Parallel Processing systems (sometimes called MPP systems), like HP and SGI/Cray, also support proprietary virtual shared-memory programming models due to the fact that these physically distributed memory systems are able to address the whole collective address space. So, for the user such systems have one *global address space* spanning all of the memory in the system. We will say a little more about the structure of such systems in section 7. In addition, packages like TreadMarks (Amza et al. 1996) provide a virtual shared-memory environment for networks of workstations.

Another trend that has came up in the last few years is *distributed processing*. This takes the DM-MIMD concept one step further: instead of many integrated processors in one or several boxes, workstations, mainframes, etc., are connected by (Gigabit) Ethernet, Fiber Channel, ATM, or otherwise and set to work concurrently on tasks in the same program.

Conceptually, this is not different from DM-MIMD computing, but the communication between processors is often orders of magnitude slower. Many packages to realize distributed computing are available. Examples of these are PVM (standing for Parallel Virtual Machine) (Geist et al. 1994), and MPI (Message Passing Interface, (Snir et al. 1998, Gropp et al. 1998)). This style of programming, called the "message passing" model has becomes so much accepted that PVM and MPI have been adopted by virtually all major vendors of distributed-memory MIMD systems and even on shared-memory MIMD systems for compatibility reasons. In addition there is a tendency to cluster shared-memory systems, for instance by HiPPI channels, to obtain systems with a very high computational power. E.g., the NEC SX-5, and the SGI/Cray SV1 have this structure. So, within the clustered nodes a shared-memory programming style can be used while between clusters message-passing should be used.

3. Shared-memory SIMD machines

This subclass of machines is practically equivalent to the single-processor vector-processors, although other interesting machines in this subclass have existed (viz. VLIW machines (van der Steen 1990)). In the block diagram in Figure 1.1 we depict a generic model of a vector architecture. The single-processor vector machine will have only one of the vector-processors depicted and the system may even have its scalar floating-point capability shared with the vector processor (as was the case in some SGI/Cray systems). It may be noted that the VPU does not show a cache. The majority of vector-processors do not employ a cache anymore. In many cases the vector unit cannot take advantage of it and execution speed may even be unfavorably affected because of frequent cache overflow.

Although vector-processors have existed that loaded their operands directly from memory and stored the results again immediately in memory (CDC Cyber 205, ETA-10), all present-day vector-processors use vector registers. This usually does not impair the speed of operations while providing much more flexibility in gathering operands and manipulation with intermediate results.

Because of the generic nature of Figure 1.1 no details of the interconnection between the VPU and the memory are shown. Still, these details are very important for the effective speed of a vector operation: when the bandwidth between memory and the VPU is too small it is not possible to take full advantage of the VPU because it has to wait for operands and/or has to wait before it can store results. When the ratio

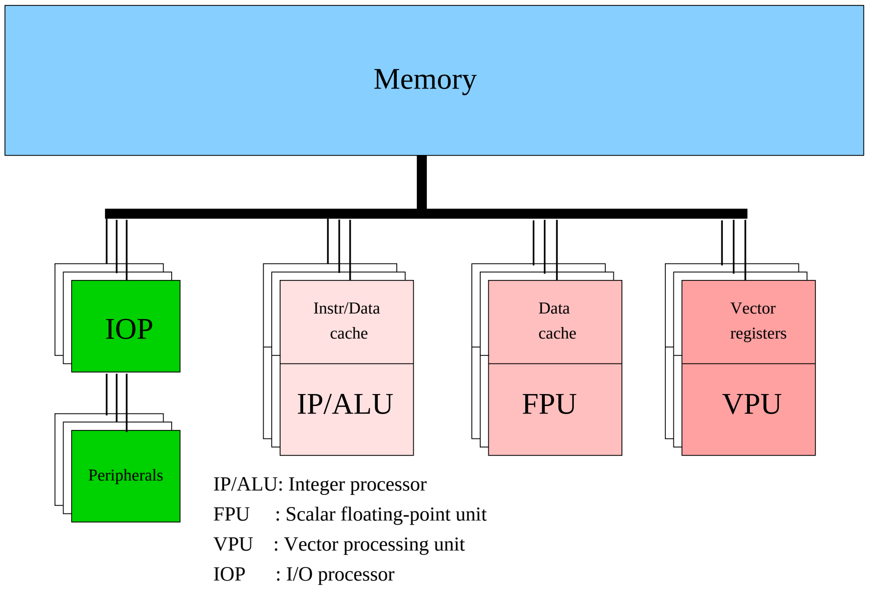

Figure 1.1. Block diagram of a vector processor.

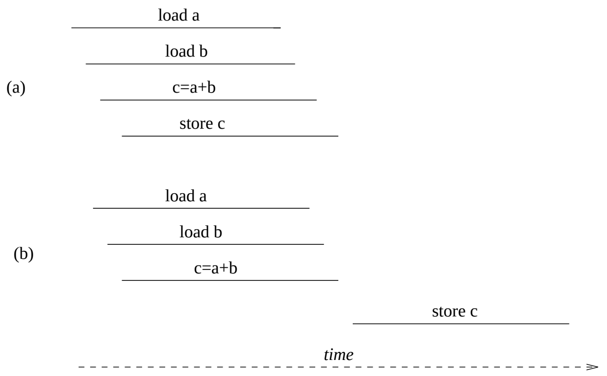

Figure 1.2. Schematic diagram of a vector addition. Case (a) when two load- and one store pipe are available; case (b) when two load/store pipes are available.

of arithmetic to load/store operations is not high enough to compensate for such situations, severe performance losses may be incurred. The influence of the number of load/store paths for the dyadic vector operation $c = a + b$ (a, b, and c vectors) is depicted in Figure 1.2. Because of the high costs of implementing these data paths between memory and the VPU, often compromises are sought and the number of systems that have the full required bandwidth (i.e., two load operations and one store operation at the *same* time) is limited. In fact, in the vector systems marketed today this high bandwidth thus not occur any longer. Vendors rather rely on additional caches and other tricks to hide the lack of bandwidth.

The VPUs are shown as a single block in Figure 1.1. Yet, again there is a considerable diversity in the structure of VPUs. Every VPU consists of a number of vector functional units, or "pipes" that fulfill one or several functions in the VPU. Every VPU will have pipes that are designated to perform memory access functions, thus assuring the timely delivery of operands to the arithmetic pipes and of storing the results in memory again. Usually there will be several arithmetic functional units for integer/logical arithmetic, for floating-point addition, for multiplication and sometimes a combination of both, a so-called compound operation. Division is performed by an iterative procedure, table look-up, or a combination of both using the add and multiply pipe. In addition, there will almost always be a mask pipe to enable operation on a selected subset of elements in a vector of operands. Lastly, such sets of vector pipes can be replicated within one VPU (2 up to 16-fold replication occurs). Ideally, this will increase the performance per VPU by the same factor provided the bandwidth to memory is adequate.

4. Distributed-memory SIMD machines

Machines of this type are sometimes also known as *processor-array* machines (Hockney and Jesshope 1987). Because the processors of these machines operate in lock-step, i.e., all processors execute the same instruction at the same time (but on different data items), no synchronization between processors is required. This greatly simplifies the design of such systems. A *control processor* issues the instructions that are to be executed by the processors in the processor array. All currently available DM-SIMD machines use a front-end processor to which they are connected by a data path to the control processor. Operations that cannot be executed by the processor array or by the control processor are offloaded to the front-end system. For instance, I/O may be through the front-end system, by the processor array machine itself or both. Figure 1.3 shows a generic model of a DM-SIMD machine of which actual models will deviate to some degree. Figure 1.3 might suggest that all processors in such systems are connected in a 2-D grid and indeed, the interconnection topology of this type of machines always includes the 2-D grid. As opposing ends of each grid line are also always connected the topology is rather that of a torus. For several machines this is not

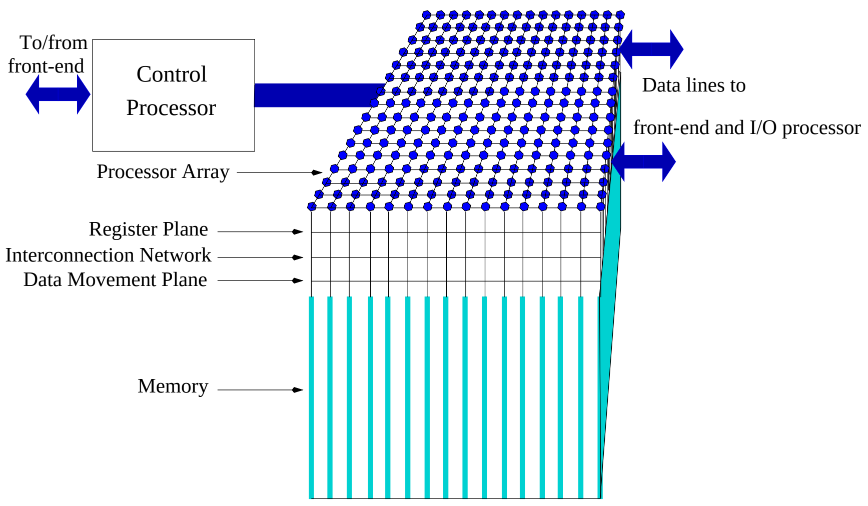

Figure 1.3. *A generic block diagram of a distributed-memory SIMD machine.*

the only interconnection scheme: They might also be connected in 3-D, diagonally, or more complex structures.

It is possible to exclude processors in the array from executing an instruction on certain logical conditions, but this means that for the time of this instruction these processors are idle (a direct consequence of the SIMD-type operation) which immediately lowers the performance. Another factor that may adversely affect the speed occurs when data required by processor i resides in the memory of processor j (in fact, as this occurs for all processors at the same time this effectively means that data will have to be permuted across the processors). To access the data in processor j, the data will have to be fetched by this processor and then send through the routing network to processor i. This may be fairly time consuming. For both reasons mentioned DM-SIMD machines are rather specialized in their use when one wants to employ their full parallelism. Generally, they perform excellently on digital signal and image processing and on certain types of Monte Carlo simulations where virtually no data exchange between processors is required and exactly the same type of operations is done on massive datasets with a size that can be made to fit comfortable in these machines.

The control processor as depicted in Figure 1.3 may be more or less intelligent. It issues the instruction sequence that will be executed by the processor array. In the worst case (that means a less autonomous control processor) when an instruction is not fit for execution on the processor array (e.g., a simple print instruction) it might be offloaded to the front-end processor which may be much slower than execution on the control processor. In case of a more autonomous control processor this can be avoided thus saving processing interrupts both on the front-end and the control processor. Most DM-SIMD systems have the possibility to handle I/O independently from the front/end processors. This is not only favorable because the communication between the front-end and back-end systems is avoided. The (specialized) I/O devices for the processor-array system is generally much more efficient in providing the necessary data directly to the memory of the processor array. Especially for very data-intensive applications like radar- and image processing such I/O systems are very important.

A feature that is peculiar to this type of machines is that the processors sometimes are of a very simple bit-serial type, i.e., the processors operate on the data items bitwise, irrespective of their type. So, e.g., operations on integers are produced by software routines on these simple bit-serial processors which takes at least as many cycles as the operands are long. So, a 32-bit integer result will be produced two times faster than a 64-bit result. For floating-point operations a similar situation holds, be it that the number of cycles required is a multiple of that needed for an integer operation. As the number of processors in this type of systems is mostly large (1024 or larger, the Alenia Quadrics is a notable exception, however), the slower operation on floating-point numbers can be often compensated for by their number, while the cost per processor is quite low as compared to full floating-point processors. In some cases, however, floating-point coprocessors are added to the processor-array. Their number is 8–16 times lower than that of the bit-serial processors because of the cost argument. An advantage of bit-serial processors is that they may operate on operands of any length. This is particularly advantageous for random number generation (which often boils down to logical manipulation of bits) and for signal processing because in both cases operands of only 1–8 bits are abundant. As the execution time for bit-serial machines is proportional to the length of the operands, this may result in significant speedups.

5. Shared-memory MIMD machines

In Figure 1.1 already one subclass of this type of machines was shown. In fact, the single-processor vector machine discussed there was a special case of a more general type. The figure shows that more than one FPU and/or VPU may be possible in one system.

The main problem one is confronted with in shared-memory systems is that of the connection of the CPUs to each other and to the memory. As

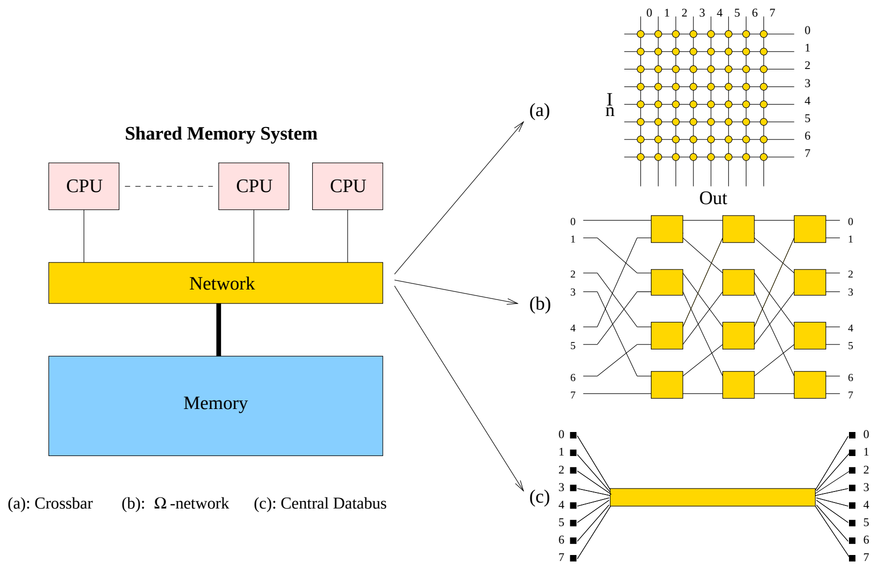

Figure 1.4. *Some examples of interconnection structures used in shared-memory MIMD systems.*

more CPUs are added, the collective bandwidth to the memory ideally should increase linearly with the number of processors, while each processor should preferably communicate directly with all others without the much slower alternative of having to use the memory in an intermediate stage. Unfortunately, full interconnection is quite costly, growing with $\mathcal{O}(n^2)$ while increasing the number of processors with $\mathcal{O}(n)$. So, various alternatives have been tried. Figure 1.4 shows some of the interconnection structures that are (and have been) used.

As can be seen from Figure 1.4, a crossbar uses n^2 connections, an Ω-network uses $n \log_2 n$ connections, while, with the central bus, there is only one connection. This is reflected in the use of each connection path for the different types of interconnections: for a crossbar each data path is direct and does not have to be shared with other elements. In case of the Ω-network there are $\log_2 n$ switching stages and as many data items may have to compete for any path. For the central data bus all data have to share the same bus, so n data items may compete at any time.

The bus connection is the least expensive solution, but it has the obvious drawback that bus contention may occur thus slowing down the computations. Various intricate strategies have been devised using caches associated with the CPUs to minimize the bus traffic. This leads however to a more complicated bus structure which raises the costs.

In practice it has proved to be very hard to design buses that are fast enough, especially where the speed of the processors has been increasing very quickly and it imposes an upper bound on the number of processors thus connected that in practice appears not to exceed a number of 10-20. In 1992, a new standard (IEEE P896) for a fast bus to connect either internal system components or to external systems has been defined. This bus, called the Scalable Coherent Interface (SCI) should provide a point-to-point bandwidth of 200-1,000 MB/s. It is in fact used in the HP Exemplar systems, but could also be used within a network of workstations. The SCI is much more than a simple bus and it can act as the hardware network framework for distributed computing, see James et al. (1990).

A multi-stage crossbar is a network with a logarithmic complexity and it has a structure which is situated somewhere in between a bus and a crossbar with respect to potential capacity and costs. The Ω-network as depicted in figure 1.4 is an example. Commercially available machines like the IBM RS/6000 SP, the SGI Origin2000, and the Cenju-4 use such a network structure, but a number of experimental machines also have used this or a similar kind of interconnection. The BBN TC2000 that acted as a virtual shared-memory MIMD system used an analogous type of network (a Butterfly-network) and it is quite conceivable that new machines may use it, especially as the number of processors grows. For a large number of processors the $n \log_2 n$ connections quickly become more attractive than the n^2 used in crossbars. Of course, the switches at the intermediate levels should be sufficiently fast to cope with the bandwidth required. Obviously, not only the *structure* but also the *width* of the links between the processors is important: a network using 16-bit parallel links will have a bandwidth which is 16 times higher than a network with the same topology implemented with serial links.

In all present-day multi-processor vector-processors crossbars are used. This is still feasible because the maximum number of processors in a system is still rather small (32 at most presently). When the number of processors would increase, however, technological problems might arise. Not only it becomes harder to build a crossbar of sufficient speed for the larger numbers of processors, the processors themselves generally also increase in speed individually, doubling the problems of making the speed of the crossbar match that of the bandwidth required by the processors.

Whichever network is used, the type of processors in principle could be arbitrary for any topology. In practice, however, bus structured machines do not have vector processors as the speeds of these would grossly mismatch with any bus that could be constructed with reasonable costs. All available bus-oriented systems use RISC processors. The local caches of the processors can sometimes alleviate the bandwidth problem if the data access can be satisfied by the caches thus avoiding references to the memory.

The systems discussed in this subsection are of the MIMD type and therefore different tasks may run on different processors simultaneously. In many cases synchronization between tasks is required and again the interconnection structure is here very important. Most vector-processors employ special communication registers within the CPUs by which they can communicate directly with the other CPUs they have to synchronize with. A minority of systems synchronize via the shared memory. Generally, this is much slower but may still be acceptable when the synchronization occurs relatively seldom. Of course in bus-based systems communication also has to be done via a bus. This bus is mostly separated from the data bus to assure a maximum speed for the synchronization.

6. Distributed-memory MIMD machines

The class of DM-MIMD machines is undoubtly the fastest growing part in the family of high-performance computers. Although this type of machines is more difficult to deal with than shared-memory machines and DM-SIMD machines. The latter type of machines are processor-array systems in which the data structures that are candidates for parallelization are vectors and multi-dimensional arrays that are laid out automatically on the processor array by the system software. For shared-memory systems the data distribution is completely transparent to the user. This is quite different for DM-MIMD systems where the user has to distribute the data over the processors and also the data exchange between processors has to be performed explicitly. The initial reluctance to use DM-MIMD machines seems to have been decreased. Partly this is due to the now existing standard for communication software (Geist et al. 1994, Snir et al. 1998, Gropp et al. 1998) and partly because, at least theoretically, this class of systems is able to outperform all other types of machines.

The advantages of DM-MIMD systems are clear: the bandwidth problem that haunts shared-memory systems is avoided because the bandwidth scales up automatically with the number of processors. Furthermore, the speed of the memory which is another critical issue with shared-memory systems (to get a peak performance that is comparable to that of DM-MIMD systems, the processors of the shared-memory machines should be very fast and the speed of the memory should match it)

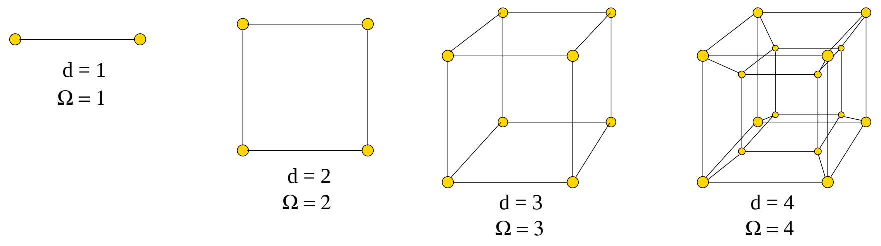

Figure 1.5. 1-, 2-, 3-, and 4-dimensional hypercube connections

is less important for the DM-MIMD machines, because more processors can be configured without the afore mentioned bandwidth problems.

Of course, DM-MIMD systems also have their disadvantages: The communication between processors is much slower than in SM-MIMD systems, and so, the synchronization overhead in case of communicating tasks is generally orders of magnitude higher than in shared-memory machines. Moreover, the access to data that are not in the local memory belonging to a particular processor have to be obtained from non-local memory (or memories). This is again on most systems very slow as compared to local data access. When the structure of a problem dictates a frequent exchange of data between processors and/or requires many processor synchronizations, it may well be that only a very small fraction of the theoretical peak speed can be obtained. As already mentioned, the data and task decomposition are factors that mostly have to be dealt with explicitly, which may be far from trivial.

It will be clear from the paragraph above that also for DM-MIMD machines both the topology and the speed of the data paths are of crucial importance for the practical usefulness of a system. Again, as in the section on SM-MIMD systems, the richness of the connection structure has to be balanced against the costs. Of the many conceivable interconnection structures only a few are popular in practice. One of these is the so-called hypercube topology as depicted in Figure 1.5.

A nice feature of the hypercube topology is that for a hypercube with 2^d nodes the number of steps to be taken between any two nodes is at most d. So, the dimension of the network grows only logarithmically with the number of nodes. In addition, theoretically, it is possible to simulate any other topology on a hypercube: trees, rings, 2-D and 3-D meshes, etc. In practice, the exact topology for hypercubes does not matter too much anymore because all systems in the market today employ what is called "wormhole routing". This means that when a message is sent from node i to node j, a header message is sent from i to j, resulting in a direct connection between these nodes. As soon as this connection is established, the data proper is sent through this connection without disturbing the operation of the intermediate nodes. Except for a small amount of time in setting up the connection between nodes, the communication time has become virtually independent of the distance between the nodes. Of course, when several messages in a busy network have to compete for the same paths, waiting times are incurred as in any network that does not directly connect any processor to all others and often rerouting strategies are employed to circumvent busy links.

A fair amount of massively parallel DM-MIMD systems seem to favor a 2- or 3-D mesh (torus) structure. The rationale for this seems to be that most large-scale physical simulations can be mapped efficiently on this topology and that a richer interconnection structure hardly pays off. However, some systems maintain (an) additional network(s) besides the mesh to handle certain bottlenecks in data distribution and retrieval (Horie et al. 1991).

A large fraction of systems in the DM-MIMD class employ crossbars. For relatively small amounts of processors (in the order of 64) this may be a direct or 1-stage crossbar, while to connect larger numbers of nodes multi-stage crossbars are used, i.e., the connections of a crossbar at level 1 connect to a crossbar at level 2, etc., instead of directly to nodes at more remote distances in the topology. In this way it is possible to connect in the order of a few thousands of nodes through only a few switching stages. In addition to the hypercube structure, other logarithmic complexity networks like Butterfly, Ω, or shuffle-exchange networks are often employed in such systems.

As with SM-MIMD machines, a node may in principle consist of any type of processor (scalar or vector) for computation or transaction processing together with local memory (with or without cache) and, in almost all cases, a separate communication processor with links to connect the node to its neighbors. Nowadays, the node processors are mostly off-the-shelf RISC processors sometimes enhanced by vector processors. A problem that is peculiar to this DM-MIMD systems is the mismatch of communication vs. computation speed that may occur when the node processors are upgraded without also speeding up the intercommunication. In some cases this may result in turning computational-bound problems into communication-bound problems.

7. CC-NUMA machines

As already mentioned in the introduction, a trend can be observed to build systems that have a rather small (up to 16) number of RISC

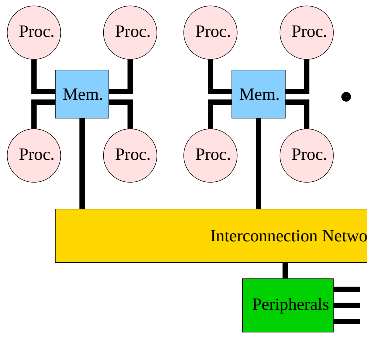

Figure 1.6. Block diagram of a system with a "hybrid" network: clusters of four CPUs are connected by a crossbar. The clusters are connected by a less expensive network, e.g., a Butterfly network

processors that are tightly integrated in a cluster, a Symmetric Multi-Processing (SMP) node. The processors in such a node are virtually always connected by a 1-stage crossbar while these clusters are connected by a less costly network. Such a system may look as depicted in Figure 1.6. Note that in Figure 1.6 all CPUs in a cluster are connected to a common part of the memory. This is similar to the policy mentioned for large vector-processor ensembles mentioned above but with the important difference that all of the processors can access all of the address space. Therefore, such systems can be considered as SM-MIMD machines. On the other hand, because the memory is physically distributed, it cannot be guaranteed that a data access operation always will be satisfied within the same time. Therefore such machines are called CC-NUMA systems where CC-NUMA stands for Cache Coherent Non-Uniform Memory Access. The term "Cache Coherent" refers to the fact that for all CPUs any variable that is to be used must have a consistent value. Therefore, it must be assured that the caches that provide these variables are also consistent in this respect. There are various ways to ensure that the caches of the CPUs are coherent. One is the *snoopy bus protocol* in which the caches listen in on transport of variables to any of the CPUs and update their own copies of these variables if they have them. Another way is the *directory memory*, a special part of memory which enables to keep track of the all copies of variables and of their validness.

For all practical purposes we can classify these systems as being SM-MIMD machines also because special assisting hardware/software (such as a directory memory) has been incorporated to establish a single system image although the memory is physically distributed.

8. Recount of (almost) available systems

In this section we give a recount of all types of systems as discussed in previous sections. When vendors market more than one type of machine we will discuss them in distinct subsections. So, for instance, we will discuss NEC systems under entries, SX-5 and Cenju-4 because they have a very different structure.

The systems are presented alphabetically. The "Machine type" entry shortly characterizes the type of system as discussed previously: Processor Array, CC-NUMA, etc.

8.1. The Alenia Quadrics

Machine type: Processor array. **Models**: Quadrics Qx, QHx, $x = 1, \ldots, 16$.
Front-end: Almost any workstation.
Operating system: Internal OS transparent to the user, Unix on front-end.
Connection structure: 3-D mesh (see remarks).
Compilers: TAO: a Fortran 77 compiler with some Fortran 90 and some proprietary array extensions.
Vendors information Web page: `www.quadrics.com`
Year of introduction: 1994.

System parameters

Model	Qx	QHx
Clock cycle	40 ns	40 ns
Theor. peak performance		
Per Proc. (32-bits)	50 Mflop/s	50 Mflop/s
Maximal (32-bits)	6.4 Gflop/s	100 Gflop/s
Main memory	$\leq$2 GB	$\leq$32 GB
No. of processors	8–128	128–2048
Communication bandwidth		
Per Proc.	50 MB/s	50 MB/s
Aggregate local	$\leq$6 GB/s	$\leq$96 GB/s
Aggregate non-local	$\leq$1.5GB/s	$\leq$24 GB/s

Remarks: The Quadrics is a commercial spin-off of the APE-100 project of the Italian National Institute for Nuclear Physics. Systems are available in multiples of 8 processor nodes in the Q-model where up to 16 boards can be fitted into one crate or in multiples of 128 nodes in the QH-model by adding up to 15 crates to the minimal 1-crate system. The interconnection topology of the Quadrics is a 3-D grid with interconnections to the opposite sides (so, in effect a 3-D torus). The 8-node floating-point boards (FPBs) are plugged into the crate backplane which provides point-to-point communication and global control distribution. The FPBs are configured as 2^3 cubes that are connected to the other boards appropriately to arrive at the 3-D grid structure.

The basic floating-point processor, the so-called MAD chip, contains a register file of 128 registers. Of these registers the first two hold permanently the values 0 and 1 to be able to express any addition or multiplication as a "normal operation", i.e., a combined multiply-add operation, where a multiplication is of the form, $a \times b + 0$ and an addition is $a \times 1 + b$. In favorable circumstances the processor can therefore deliver two floating-point operations per cycle. Instructions are centrally issued by the controller at a rate of one instruction every two clock cycles.

Communication is controlled by the Memory Controller and the Communication Controller which are both housed on the backplane of a crate. When the Memory Controller generates an address it is decoded by the Communication Controller. In case non-local access is desired, the Communication Controller will provide the necessary data transmission. The memory bandwidth per processor is 50 MB/s which means that every 2 cycles an operand can be shipped into or out of a processor. The bandwidth for non-local communication turns out to be only four times smaller than local memory access.

The Quadrics communicates with the front-end system via a T805 transputer-based interface system, called the Local Asynchronous Interface (LAI). The interface can write and read the memories of the nodes and the Controller. Presently, the bandwidth of the interface to the front-end processor is not very large (1 MB/s). It is expected that this can be improved by about a factor of 30 in the near future. I/O has to be performed via the front-end system and will therefore be relatively slow.

The TAO language has several extensions to employ the SIMD features of the Quadrics. Firstly, floating-point variables are assumed to be local to the processor that owns them, while integer variables are assumed to be global. Local variables can be promoted to global variables. Other extensions are the `ANY`, `ALL`, and `WHERE/END WHERE` keywords that can be used for global testing and control. Processors that not meet a global condition effectively skip the operation(s) that are associated with it. For easy referencing nearest-neighbor locations special constants `LEFT`, `RIGHT`, `UP`, `DOWN`, `FRONT`, and `BACK` are provided. In addition, new data types and operators on these data types are supported together with overloading of operators. This enables very concise code for certain types of calculations.

Measured performances: No measured performances have been reported for this machine.

8.2. The Avalon A12

Machine type: RISC-based distributed-memory multi-processor.
Models: Avalon A12.
Operating system: AVALON micro kernel based Unix (Image compatible with Digital Unix).
Connection structure: Multistage variable (see remarks).
Compilers: Fortran 77, Fortran 90 extensions, HPF, ANSI C.
Vendors information Web page: `www.teraflop.com/`
Year of introduction: 1996.

System parameters

Model	A12
Clock cycle	2.5 ns
Theor. peak performance	
Per Proc. (64-bits)	800 Mflop/s
Maximal	1.3 Tflop/s
Memory/node	$\leq$ 1 GB
Memory/maximal	1.7 TB
No. of processors	12–1680
Communication bandwidth	
Point-to-point	128–400 MB/s
Bisectional (maximal)	10 GB/s

Remarks: The A12 is be based on the DEC Alpha 21164 RISC processor. The processor used in the system has a clock cycle of 2.5 ns. However, most of the information given at the vendors Web page still uses the data for a node processor with a 3.3 ns clock to describe the configuration properties. The Web information is therefore internally somewhat inconsistent. Because the Alpha 21164 has dual floating-point arithmetic pipes it will deliver a theoretical peak performance of 800 Mflop/s. The maximum configuration of the system is given as 1680 processors The first and second level cache reside on chip, a 1 MB third level cache is provided on each A12 CPU card. The bandwidth to/from the first level cache is sufficient to transport two operands to the CPU and to ship one result back in one cycle. The second level cache has two-thirds of this bandwidth, while the third level cache has the capability of providing an 64-bit word every two cycles. The bandwidth to/from memory is 400 MB/s or one 64-bit word every 8 cycles. The memory has two-way interleaved banks of a memory that can be up to 1 GB/node.

Each CPU card contains a Alpha 21164 processor, the third level or B cache and the local memory for that node. Twelve CPU cards can be housed in one crate which has a full crossbar backplane. This yields a inter-node bandwidth of slightly under 400 MB/s between the cards within one crate. Apart from the 12 slots for CPU cards, there are two extra dual channel slots that can accommodate communication cards that provide the connections with other crates. For the in-crate crossbar CMOS technology is used. However, for the inter-crate connections ECL logic is employed. The actual connections between crates are made by coaxial cables. This way of connection provides a large flexibility in the overall interconnection topology: one could build trees or toruses or a secondary level crossbar (in the last case one crate should be filled entirely with communication cards to build a 144 processor system). The communication speed between crates is less fast (but still respectable): 128 MB/s. Various configurations are described at the Web-address given above.

I/O can be configured in various ways: It is possible to put 32-bit or 64-bit PCI expansion cards on each CPU card to obtain what Avalon calls "Type 1 I/O nodes". Also, a direct switch connection via a variant of the communication card can be made to the outside world. Depending on the number of cards the bandwidth is 400 or 800 MB/s for this type 3 I/O node. The type 2 I/O node is in fact a dedicated TCP/IP connection as needed for the control workstation required by the system.

Measured Performances: A 140-node A12 was installed by the end of 1996 at Los Alamos National Laboratory of which the processors had a faster clock: 1.88 ns, instead of 2.5 ns. In Dongarra (1999) a speed of 48.6 Gflop/s was reported for this configuration on the solution of a full linear system of order 62720, 33% of the Theoretical Peak Performance.

8.3. The Cambridge Parallel Processing Gamma II

Machine type: Processor array.
Models: Gamma II Plus 1000, Gamma II Plus 4000.
Front-end: DEC, HP, or Sun workstation, stand-alone for dedicated applications.
Operating system: Internal OS transparent to the user, Unix on front-end.
Connection structure: 2-D mesh, row- and column data paths (see remarks).
Compilers: FORTRAN-PLUS (a Fortran 77 compiler with some Fortran 90 and some proprietary array extensions), C++.

Vendors information Web page: `www.cppus.com`
Year of introduction: 1995.

System parameters

Model	Gamma II Plus 1000	Gamma II Plus 4000
Clock cycle	33 ns	33 ns
Theor. peak performance		
Per Proc. (32-bits)	0.6 Mflop/s	0.6 Mflop/s
Maximal (32-bits)	0.6 Gflop/s	2.4 Gflop/s
1-bit (Gop/s)	30.7	122.8
8-bit (Gop/s)	30.7	122.8
Program memory	$\leq$4 MB	$\leq$4 MB
No. of processors	$\leq$128 MB	$\leq$512 MB
Internal communication speed		
Across row, column	120 MB/s	480 MB/s
Memory to PE	3.84 GB/s	15.4 GB/s

Remarks: In November 1995 the new Gamma II Plus models have been announced by CPP. In essence there is not much difference with its predecessor the DAP Gamma. However, the clock cycle has tripled to 33 ns with an equivalent rise in the peak performance of the systems.

The Gamma II is presented as the fourth generation of this type of machine. Indeed, the macro architecture of the systems has hardly changed since the first ICL DAP (the first generation of this system) was conceived. As in the ICL DAP in the Gamma 1000 models the 1024 processors are ordered in a 32×32 array, while the Gamma 4000 has 4096 processors arranged in a 64×64 square.

The systems are able to operate byte parallel on appropriate operands to speed up floating-point operations, however, for logical operations bit-wise operations are possible, which makes the machines quite fast in this respect. As the byte parallel code consists of separate sequences of microcode instructions, the bit processor plane and the byte processor plane are in fact independent and can work in parallel. This is also the case for I/O operations. Also character-handling can be done very efficiently. This is the reason that Gamma systems are often used for full text searches.

As in all processor-array machines, the control processor (called the Master Control Unit (MCU) in the DAP) has a separate memory to hold program instructions while the data are held in the data memory associated with each Processing Element (PE) in the processor array. So, for a Gamma 1000 with 128 MB of data memory each PE has 128 KB of data memory directly associated to it. To access data in other

PEs memories these must be brought up to the data routing plane and shifted to the appropriate processor.

As already mentioned under the heading of the connection structure, there are two ways of connecting the PEs. One is the 2-D mesh that connects each element to its North-, East-, West-, and South neighbor. In addition there are row- and column data paths that enable the fast broadcast of a row or column to an entire matrix by replication. Conversely, they can be used for row or column-wise reduction of matrix objects into a column or row-vector of results from, e.g., a summing or maximum operation.

Separate I/O processors and disk systems can be attached to the Gamma directly thus not burdening the front-end machine (and the connection between front-end and Gamma) with I/O operations and unnecessary data transport. One of these I/O devices is the GIOC that can transport data to the data memory at a sustained rate of 80 MB/s transposing the data to the vertical storage mode of the data memory on the fly. Also, a direct video interface is available to operate a frame buffer.

A nice (non-standard) feature of the FORTRAN-PLUS compiler is the possibility to use logical matrices as indexing objects for computational matrix objects. This enables a very compact notation for conditional execution on the processor array. Since 1997 also C++ is available.

Measured Performances: In Flanders (1991) the speed of matrix multiplication on various DAP models (precursors of the Gamma systems) is analyzed. The documentation states 32-bit floating-point add speed of 1.68 Gflop/s on 4096 PEs, while a 32-bit 1,024 complex FFT would attain 2.49 Gflop/s. No independent performance figures for the Gamma II Plus systems are available.

8.4. The C-DAC PARAM OpenFrame system

Machine type: RISC-based distributed-memory multi-processor.
Models: PARAM OpenFrame 9000 system.
Operating system: PARAS micro kernel based Unix (compatible with Sun's Solaris).
Connection structure: Multistage variable (see remarks).
Compilers: Fortran 77, Fortran 90, HPF, ANSI C, C++.
Vendors information Web page: `www.soft.net/cdac/`
Year of introduction: 1996.

System parameters

Model	OpenFrame 9000
Clock cycle	3 ns (see remarks)
Theor. peak performance	
Per Proc. (64-bits)	600 Mflop/s (see remarks)
Maximal	600 Gflop/s (see remarks)
Memory/node	—
Memory/maximal	—
No. of processors	1–1024
Communication bandwidth	
Point-to-point	80 MB/s
Aggregate bandwidth	3.2 GB/s

Remarks: The OpenFrame 9000 system is the fourth generation of CDAC machines that is developed by CDAC, the Centre for Development of Advanced Computing, an institute in India that has as its mission to develop an manufacture "state-of-the-art open architecture supercomputers". This system is the second generation that is marketed abroad. In the predecessor, the PARAM 9000/SS SuperSPARC II processors were used. In the present model Sun UltraSPARCs or DEC/Compaq Alpha chips are employed or even mixtures of these. Also, the maximum possible number of processors has been increased from 200 to 1024 in replicatable units of 32 processors. As the type of processor and the clock cycle are not fixed, no theoretical peak performance can be specified. When the same basic processors are assumed as Sun employs in most of its Enterprise servers, the estimates as given in the parameter list above seem more or less indicative. The documentation also does not reveal any details of the type of the interconnection network except that cut-through wormhole routing is used. A point-to-point communication bandwidth of 80 MB/s (bi-directional) is quoted. The aggregate bandwidth for a maximal configuration is 3.2 GB/s.

The amount of available software shows that the PARAM OpenFrame 9000 is not a first-generation system. Apart from Fortran 77, Fortran 90, HPF, and C++ are available and the CORE, MPI, and PVM message passing interfaces are available. There is a parallel debugger, a proprietary performance evaluation tool called AIDE, while TOTALVIEW can be delivered on request.

In addition, a library of parallel routines, PARUL, is available. This library contains PVM versions of dense linear algebra routines, eigenvalue routines, and FFTs.

Measured Performances: No measured performances of the PARAM OpenFrame 9000 are available at this moment for any configuration.

8.5. The Compaq/DEC GS60/140

Machine type: RISC-based shared-memory multiprocessor.
Models: GS60, GS140, Cluster.
Operating system: Digital Unix (DEC's flavor of Unix).
Compilers: Fortran 77, HPF, C, C++.
Vendors information Web page: `www.digital.com/info/hpc`
Year of introduction: 1998.

System parameters

Model	GS60	GS140	Cluster
Clock cycle	1.66 ns	1.66 ns	1.66 ns
Theor. peak performance			
Per Proc.	1.2 Gflop/s	1.2 Gflop/s	1.2 Gflop/s
Maximal	7.2 Gflop/s	16.8 Gflop/s	67.2 Gflop/s
Memory	$\leq$ 12 GB	$\leq$ 28 GB	$\leq$ 112 GB
No. of processors	$\leq$ 6	$\leq$ 14	$\leq$ 56
Memory bandwidth			
Processor/memory	1.87 GB/s	1.87 GB/s	1.87 GB/s
Between cluster nodes	—	—	100 MB/s

Remarks: The GS60 and GS140 are almost identical to their predecessors, the AlphaServers 8200 and 8400. The difference lies in the processor that is used: instead of the Alpha 21164 at a clock rate of 1.6 ns in the new systems an Alpha 21264 with a clock rate of 1.66 ns is used. Note that this leads to a *decrease* in the theoretical peak performance from 1.25 to 1.2 Gflop/s per processor. However, Compaq claims that the new processor will generally give a performance *increase* of a factor 2.5 with respect to the Alpha 21164 which is not unlikely with the improvements made in the chip.

The GS60 and GS140 are symmetric multi-processing systems. The GS60 model is a somewhat smaller copy of the GS140 model: in the GS60 a maximum of 6 CPUs can be accommodated while this number is 14 for the GS140 model. Also, there is room for at most 12 GB of memory in the GS60 while the GS140 can house 28 GB. However, the amount of CPUs and memory is not independent. For instance, the GS140 has 9 system slots. One of these is reserved for I/O and one will have to contain at least one CPU module which can contain 1 or 2 CPUs. From the remaining slots 7 can be used either for memory or for a CPU module. So, one has to choose for either higher computational power or for more memory. This can potentially be a problem for large applications that require both.

The GS systems (GS stands for Global Server) can be clustered using PCI bus MemoryChannel link cables that are connected to a hub. The systems need not be of the same model. The bandwidth of this interconnect is slightly over 100 MB/s. Up to four systems can be coupled in this way. To support this kind of cluster computing, HPF and optimized versions of PVM and MPI are available.
Measured Performances: For the the GS60 and GS140 no performance figures are available yet.

8.6. The Fujitsu AP3000

Machine type: RISC-based distributed-memory multi-processor.
Models: AP3000.
Operating system: Cell OS (transparent to the user) and Solaris (Sun's Unix variant) on the front-end system.
Connection structure: 2-D torus.
Compilers: Parallel Fortran/AP, Fortran 90, HPF, C, C++.
Vendors information Web page: `www.fujitsu.com`
Year of introduction: 1996.

System parameters

Model	AP3000
Clock cycle	3.3 ns
Theor. peak performance	
Per Proc. (64-bits)	600 Mflop/s
Maximal	614 Gflop/s
Memory/node	$\leq$ 2 GB
Memory/maximal	$\leq$ 2 TB
No. of processors	4–1024
Communication bandwidth	
Point-to-point	200 MB/s

Remarks: The AP3000 is the successor of the earlier AP1000 system. Although the name could suggest otherwise, few characteristics of the AP1000 have been retained except that Sun SPARC processors are used in the nodes. No front-end processor is required anymore as in the former system.

Also the communication network has been simplified considerably with respect to that in the earlier model: where three different networks were present in the AP1000 (see Horie et al. (1991)), in the AP3000 the nodes are connected in a 2-D torus structure with a bi-directional bandwidth of 200 MB/s and there is a separate control network. The maximum amount of memory is huge: a full 1024 node system can accommodate 2 TB.

Another difference with the AP1000 system is that the fastest nodes (the U300 nodes described here) can have either 1 or 2 CPUs as opposed to only one CPU in the AP1000. The two CPUs share the on-board memory.

The available software for the AP3000 is extensive: Parallel Fortran/AP is a Fortran 77 with extensions that offers a shared-memory-like programming model for the system. In addition, HPF is available and the machine can also be used with a message passing model as customized MPI/AP and PVM/AP are offered. As sequential languages to be used with the message passing libraries Fortran 90, C and C++ are available.

Measured Performances: The system has been announced in March 1996 and installations have been done in Japan, the University of Singapore and at the Australian National University but as yet no performance figures are published.

8.7. The Fujitsu VPP700 series

Machine type: Distributed-memory vector multi-processor.
Models: VX-E, VPP300-E, VPP700-E.
Operating system: UXP/V (a V5.4 based variant of Unix).
Connection structure: Full distributed crossbar.
Compilers: Fortran 90/VP (Fortran 90 Vector compiler), Fortran 90/ VPP (Fortran 90 Vector Parallel compiler), C/VP (C Vector compiler), C, C++.
Vendors information Web page: `www.fujitsu.com`
Year of introduction: VX, VPP300: 1995, VPP700: 1996.

System parameters

Model	VX-E	VPP300-E	VPP700-E
Clock cycle	6.6 ns	6.6 ns	6.6 ns
Theor. peak performance			
Per Proc. (64-bits)	2.4 Gflop/s	2.4 Gflop/s	2.4 Gflop/s
Maximal	9.6 Gflop/s	38.4 Gflop/s	614.4 Gflop/s
Memory/node	$\leq$ 2 GB	$\leq$ 2 GB	$\leq$ 2 GB
Memory/maximal	$\leq$ 8 GB	$\leq$ 32 GB	$\leq$ 512 GB
No. of processors	1–4	1–16	8–256
Memory bandw./proc.	19.6 GB/s	19.6 GB/s	19.6 GB/s
Communication bandwidth			
Point-to-point	615 MB/s	615 MB/s	615 MB/s

Remarks: The VX-E, VPP300-E, and VPP700-E systems (with E for extended) are “midlife kickers”: minor extensions of the VX, VPP300, and VPP700. The only difference is a slightly faster system clock: 6.6 ns in the E models instead of the 7 ns in the former systems. There are no architectural changes. The VPP300 is a successor to the earlier VPP500. It is a much cheaper CMOS implementation of its predecessor with some important differences. First, no VPX200 front-end system is required anymore. Second, the crossbar that is used to connect the vector nodes is distributed. Therefore, the cost of a system is scalable: one does not need to buy a complete enclosure with the full crossbar for only a few nodes. The VX series is in fact a smaller version of the VPP300 with a maximum of 4 processors. Both the VX machines and the VPP300 systems are air-cooled.

The architecture of the VPP300 nodes is almost identical to that of the VPP500: Each node, called a Processing Element (PE) in the system is a powerful (2.4 Gflop/s peak speed with a 6.6 ns clock) vector processor in its own right. The vector processor is complemented by a Large Instruction Word scalar processor with a peak speed of 300 Mflop/s. The scalar instruction format is 64 bits wide and may cause the execution of three operations in parallel. Each PE has a memory of up to 2 GB while a PE communicates with its fellow PEs at a point-to-point speed of 570 MB/s. This communication is cared for by separate Data Transfer Units (DTUs). To enhance the communication efficiency, the DTU has various transfer modes like contiguous, stride, sub array, and indirect access. Also translation of logical to physical PE-ids and from Logical in-PE address to real address are handled by the DTUs. When synchronization is required each PE can set its corresponding bit in the Synchronization Register (SR). The value of the SR is broadcast to all PEs and synchronization has occurred if the SR has all its bits set for the relevant PEs. This method is comparable to the use of synchronization registers in shared-memory vector processors and much faster than synchronizing via memory.

The VPP700 is a logical extension of the VPP300. While the processors in the latter machine are connected by a full crossbar, the maximum configuration of a VPP700 consists of 16 clusters of 16 processors connected by a level-2 crossbar. So, a fully configured VPP700 consists in fact of 16 full VPP300s. Because the diameter of the network is 2 (for the larger configurations) instead of 1 as in the VPP300, the communication time between processors will be slightly larger. At the moment this worst case increase is not exactly known to the author.

The Fortran compiler that comes with the VPP300/700 has extensions that enable data decomposition by compiler directives. This evades in many cases restructuring of the code. The directives are different from those as defined in the High Performance Fortran Proposal but it should be easy to adapt them. Furthermore, it is possible do define parallel regions, barriers, etc., via directives, while there are several intrinsic functions to enquire about the number of processors and to execute `POST/WAIT` commands. Furthermore, also a message passing programming style is possible by using the PVM or PARMACS communication libraries that are available. Of course the software for the VPP700 and the VPP300 is exactly the same and the systems can run each others executables.

Measured Performances: Of the VX-E, VPP300-E, and VPP700-E no performance figures are known but in Dongarra (1999) results for the VX, the VPP300, and the VPP700 are given. The speed for solving dense linear system of sizes 28,800 59,200, and 111,360 was 8.6, 34.1, and 213 Gflop/s on a 4 proc. VX, a 16 proc. VPP300, and a 116 proc. VPP700, respectively.

8.8. The Hitachi S3600 series

Machine type: Vector-processor
Models: S3600/120, S3600/140, S3600/160, S3600/180
Operating system: VOS3/HAP/ES (IBM MVS compatible) and OSF/1
Compilers: FORT77/HAP vectorizing Fortran 77, C, C++.
Vendors information Web page: none.
Year of introduction: 1994.

System parameters

Model	S3600/120	S3600/140	S3600/160	S3600/180
Clock cycle VPU	4 ns	4 ns	4 ns	4 ns
Clock cycle scal. proc.	8 ns	8 ns	8 ns	8 ns
Theor. peak perform.	0.25 Gflop/s	0.25 Gflop/s	1.0 Gflop/s	2.0 Gflop/s
Main memory	128-256 MB	256-512 MB	256-512 MB	512-1024 MB
Extended memory	$\leq$ 6 GB	$\leq$ 16 GB	$\leq$ 16 GB	$\leq$ 16 GB

Remarks: The S3600 system is the only single-CPU vector-processor that is still marketed and it might be withdrawn soon in favor of the Hitachi SR8000 (see below).

The speed differences between the different models stem from replication of the multiply/add pipe in the models S3600/120-180. The /160 and /180 models have respectively two- and four-fold sets of a separate add- and a multi-functional multiply/add vector pipes. This should lead to a maximum of 3 results per clock cycle per pipe set. So, contrary to the information given by the vendor, the maximum performance of, e.g., the /180 should in some situations be 3 Gflop/s instead of 2.

All configurations of the S3600, as in its direct predecessor the S-820, are air cooled while most machines in this class rely at least on water cooling.

Unlike the S-820 series, the S3600 series is also marketed worldwide, and not only in Japan.

Measured performances: In Dongarra (1999) a speed of 851 Mflop/s for the solution of a full linear system of order 1000 is reported for the S3600/160. The S3600/180 attains a performance of 1672 Mflop/s on the same problem.

8.9. The Hitachi SR8000

Machine type: RISC-based distributed-memory multi-processor.
Models: SR8000.
Operating system: HI-UX/MPP (Micro kernel Mach 3.0).
Connection structure: Multi-dimensional crossbar (see remarks).
Compilers: Fortran 77, Fortran 90, Parallel Fortran, HPF, C, C++.
Vendors information Web page:
`www.hitachi.co.jp/Prod/comp/hpc/eng/sr81e.html`
Year of introduction: 1998.

System parameters

Model	SR8000
Clock cycle	4.0 ns
Theor. peak performance	
Per Proc. (64-bits)	8 Gflop/s
Maximal	1 Tflop/s
Memory/node	$\leq$ 8 GB
Memory/maximal	$\leq$ 1 TB
No. of processors	4–128
Communication bandwidth	
Point-to-point	1 GB/s

Remarks: The SR8000 is the third generation of distributed-memory parallel systems of Hitachi. It is to replace both its direct predecessor, the SR2201 and the late top-vector-processor, the S-3800 (see 9).

The basic node processor is a 4 ns clock PowerPC node with major enhancements made by Hitachi. E.g., a hardware barrier synchronization is added and the additions required for "Pseudo Vector Processing" (PVP). The latter means that for operations on long vectors one does not incur the detrimental effects of cache misses that often ruin the performance of RISC processors unless code is carefully blocked and unrolled. This facility was already available on the SR2201 and experiments have shown that this idea seems to work well (see Hit (2001)).

The peak performance per basic processor, or IP, can be attained with 2 simultaneous multiply/add instructions resulting in a speed of 1 Gflop/s. However, eight basic processors are coupled to form one processing node all addressing a common part of the memory. For the user this node is the basic computing entity with a peak speed of 8 Gflop/s. Hitachi refers to this node configuration as COMPAS, Co-operative Micro-Processors in single Address Space. In fact this is a kind of SMP clustering as discussed in sections 2 and 7. A difference with most of these systems is that for the user the individual processors in a cluster node are not accessible. Every node also contains an SP, a system processor that performs system tasks, manages communication with other nodes and a range of I/O devices.

The SR8000 has a multi-dimensional crossbar with a bi-directional link speed of 1 GB/s. From 4–8 nodes the cross-section of the network is 1 hop. For configurations 16–64 it is 2 hops and for a 128-node system it is 3 hops.

Like in some other systems as the SGI/Cray T3E (8.17), Meiko CS-2 (8.12), and the NEC Cenju-4 (8.13), one is able to directly access the memories of remote processors. Together with the fast hardware-based barrier synchronization this should allow for writing distributed programs with very low parallelization overhead.

The following software products are supported in addition to those already mentioned above: PVM, MPI, PARMACS, Linda, and FORGE90. In addition several numerical libraries like NAG and IMSL are offered.

Measured Performances: As of January 1999 the first installations were planned. A maximal configuration has just been placed at the University of Tokyo Computing Centre. At this moment no performance figures are available.

8.10. The HP Exemplar V2500

Machine type: RISC-based CC-NUMA system.
Models: Exemplar V2500.
Operating system: HP-UX (HP's usual Unix flavor).
Connection structure: Ring.
Compilers: Fortran 77, Fortran 90, Parallel Fortran, HPF, C, C++.
Vendors information Web page: `www.hp.com`
Year of introduction: 1998.

System parameters

Model	Exemplar V2500
Clock cycle	2.27 ns
Theor. peak performance	
Per Proc. (64-bits)	1.76 Gflop/s
Maximal	225.3 Gflop/s
Memory/node	≤1 GB
Memory/maximal	≤16 GB
No. of processors	2–128
Communication bandwidth	
Aggregate (per cabinet)	15.36 GB/s
Aggregate (inter-cabinet)	3.84 GB/s

Remarks: The V2500 is the latest in the series of Exemplar systems that have been offered first by Convex and later by HP since 1995 (see section 9). The architecture, however, has not radically changed: up to 32 PA-RISC 8500 chips are clustered via a crossbar to form an SMP node. The PA-RISC 8500 CPUs run at a clock cycle of 2.27 ns. As a CPU contains 2 floating-point units that are able do execute a combined floating multiply-add instruction, in favorable circumstances four flops/cycle can be achieved and a Theoretical Peak Performance of 1.76 Gflop/s per CPU can be attained. Per SMP node the peak speed is 56.32 Gflop/s.

Up to four SMP nodes can be coupled by a so-called SCA HyperLink, uni-directional SCI rings with an aggregate bandwidth of 3.84 GB/s, while the aggregate bandwidth within an SMP node is 15.36 GB/s. The HyperLinks tolerate multiple outstanding requests and, in addition, there is a "HyperLinkcache" that both help in hiding the communication latency in inter-node communication.

As in the former systems a shared memory parallel model is supported. HP is a partner in the OpenMP organization and will therefore make available this style of shared-memory parallel programming in addition to (and later on instead of) its proprietary parallel model. The shared-memory parallelism is not confined to the SMP nodes: a multi-node system can be addressed globally making the Exemplar a CC-NUMA system. The memory latency within and between nodes differs by about a factor of 3–3.5.

Measured Performances: In Dongarra (1999) a speed of 31.59 Gflop/s is reported for a 1-cabinet, 32 processor system when solving a 41,000-order dense linear system, an efficiency of 56% on this problem.

8.11. The IBM RS/6000 SP

Machine type: RISC-based distributed-memory multi-processor.
Models: IBM RS/6000 SP.
Operating system: AIX (IBM's Unix variant).
Connection structure: Ω-switch.
Compilers: XL Fortran (Fortran 90), HPF, XL C, C++.
Vendors information Web page:
`www.rs6000.ibm.com/hardware/largescale/index.html`.
Year of introduction: 1998 (POWER3 SMP), 1997 (P2SC).

System parameters

Model	RS/6000 SP POWER3 SMP	RS/6000 SP P2SC
Clock cycle	3 ns	6.25 ns
Theor. peak perform.		
Per Proc. (64-bits)	666 Mflop/s	640 Mflop/s
Maximal	variable (see remarks)	variable (see remarks)
Memory/node	$\leq$4 GB	$\leq$1/2 GB (see remarks)
Memory/maximal	$\leq$0.5 TB	$\leq$1 TB
No. of processors	8–512	8–512
Comm. bandwidth		
Point-to-point	160 MB/s	160 MB/s

Remarks: The variety in the types of nodes that are available for the RS/6000 SP is short of bewildering (IBM provides a 48 page document for selecting the appropriate nodes in an SP system). We only discuss the subset that is most relevant for scientific and technical computation, i.e., the P2SC thin nodes and the POWER3 SMP thin and wide nodes. The P2SC nodes can deliver 4 floating-point results per clock cycle. As the fastest of the P2SC nodes has a clock cycle of 6.25 ns, it has a peak performance of 640 Mflop/s while a single POWER3 processor can attain 666 Mflop/s at maximum with 2 floating-point units. Another difference is that the P2SC nodes have a primary data cache of 128 KB while it is only 64 KB on the POWER3 chip. On the other hand, the P2SC has no secondary cache while the POWER3 has an up to 16 MB secondary cache.

IBM positions the P2SC-based and POWER3 systems primarily for the technical/scientific market. In the parameter list above we included the presently fastest P2SC and the POWER3 processors. POWER3s can be combined into a 2-way SMP cluster with a peak performance of 1.33 Gflop/s.

The SP configurations are housed in columns, "tall" or "short" frames, of which the tall frames can contain 8–16 processor nodes and short frames half of the tall frames. How many actually are installed depends on the type of node employed: a thin node occupies half of the space of a wide node. Although the processors in these nodes are basically the same there are some differences. At the time of writing no 6.25 ns clock P2SC wide nodes were available yet. The fastest in this class feature a clock cycle of 7.4 ns giving a peak speed of 540 Mflop/s. For the POWER3 nodes there is no difference in speed between thin and wide nodes. Each frame is recommended to be configured with at least one wide node, although a frame completely filled with thin nodes seems possible according to the documentation.

POWER3 wide nodes have 10 PCI slots against only 2 PCI slots in the thin node. The P2SC-based nodes use MicroChannel instead of PCI busses and wide nodes have a double amount of MicroChannel slots (8 instead of 4) as compared to the thin nodes. Furthermore, the maximum memory of a P2SC wide node can be 2 GB whereas the maximum for thin nodes is 1 GB. For POWER3 nodes there is no difference between thin and wide nodes with respect to the maximum amount of memory. IBM envisions the wide node more or less as a server for a frame and recommends configurations of one wide node packaged with 14 thin nodes per column (although this may differ with the needs of the user). The RS/6000 SP is accessed through a front-end control workstation that also monitors system failures. Failing nodes can be taken off line and exchanged without interrupting service. In addition, file servers can be connected to the system while every node can have up to 2 GB. This can greatly speed up applications with significant I/O requirements.

The so-called high-performance switch that connects the nodes is an Ω-switch as described in section 5 and, although we mentioned only the highest speed option for the communication, the high-performance switch, there is a wide range of other options that could be chosen instead: Ethernet, Token Ring, FDDI, etc., are all possible. The best measured speeds of the high-performance switch are about 110 MB/s in point-to-point communication while a bandwidth for the communication ports of 160 MB/s is quoted for P2SC nodes and of 480 MB/s for POWER3 nodes. Unfortunately, the online (semi)technical information of IBM is not very helpful in providing more detailed information with regard to the other properties of the switch so, for instance, a bisection bandwidth cannot be provided at this point. The high-performance switch has some redundancy built into it for greater reliability.

Applications can be run using PVM or MPI. Also High Performance Fortran is supported, both a proprietary version and a compiler from the Portland Group. IBM uses its own PVM version from which the data format converter XDR has been stripped. This results in a lower overhead at the cost of generality. Also the MPI implementation, MPI-F, is optimized for the RS/6000 SP systems.

Commercially, systems up to 512 nodes are marketed, but larger systems are possible. In fact, the POWER3 node is a first commercial spin-off of the ASCI Blue Pacific system with more than 1300 processors (see ASCI (2001)).

Measured Performances: In Dongarra (1999) a performance of 547.0 Gflop/s for a 475 604e based node (1900 processor) system is reported for solving a 244000-order dense linear system, while a POWER3 based system with 1344 processors attained a speed of 468.2 Gflop/s on a similar problem of order 205000. This amounts to efficiencies of 43 and 52%, respectively.

8.12. The Meiko Computing Surface 2

Machine type: Distributed-memory multi-vector-processor.
Models: Computing Surface 2.
Operating system: Internal OS transparent to the user, Solaris (Sun's Unix variant) on the front-end system.
Connection structure: Multistage crossbar.
Compilers: Extended Fortran 77, ANSI C.
Vendors information Web page: `www.meiko.com`.
Year of introduction: 1994.

System parameters

Model	Computing Surface 2
Clock cycle	20 ns
Theor. peak performance	
Per Proc. (64-bits)	200, 40 Mflop/s
Maximal	204.8 Gflop/s
Memory/node	32–128, 32–512 MB
Memory/maximal	≤128 GB
No. of processors	8–1024
Communication bandwidth	
Point-to-point (bi-directional)	50 MB/s

Remarks: The CS-2 features 8-1,024 processor elements (PEs) which can be either scalar or vector nodes. Apart from a separate communications module, these PEs contain either a SuperSPARC or a SuperSPARC + 2 μVP vector-processors. The speed of a scalar PE is estimated to be 40 Mflop/s (at a 20 ns clock) and 200 Mflop/s for the vector PEs for 64-bit precision. The μVP modules are manufactured by Fujitsu. The speed at 32-bit precision is doubled with respect to 64-bit operation and, unlike the early Fujitsu VP products, use IEEE 754 floating-point format. The memory has 16 banks and to avoid memory bank conflicts the CS-2 has the interesting option to have scrambled allocation of addresses, thus guaranteeing good access at potential problematic strides 2, 4, etc.

The point-to-point communication speed is 100 MB/s (50 MB/s in each direction). Because the communication happens through multilevel crossbars, called "layers" by Meiko, the aggregate bandwidth of the system scales with the number of PEs, with a latency of 200 ns per layer. As the maximum configuration of the machine contains 1,024 PEs, the theoretical peak performance at 64-bit precision is about 200 Gflop/s. It is possible to connect each PE to its own I/O devices to have scalable parallel I/O with the scaling of other resources.

The Portland Group which has won some renown for its excellent i860 compilers has developed the compilers for the CS-2. These include Fortran 77, Fortran 90, and ANSI C. The compiler offers HPF data distribution directives. Furthermore, some optimized standard linear algebra and FFT routines are offered via a proprietary numerical library.

In the USA the machine is marketed by Meiko. In 1996 Meiko has merged with Alenia, the same firm that also markets the Alenia Quadrics. Although the new marketing policy never has been made clear, it may be assumed that Alenia will market the system in Europe and the rest of the world (see `www.quadrics.com`).

Measured Performances: In Dongarra (1999) a speed of 5.0 Gflop/s on a 64 processor CS-2 is reported for the solution of an order 18688 dense linear system. From the NAS parallel benchmarks (NPB 1997) some results on a 128 processor machine are given for class B problems: EP took 21.16 seconds while 6.52 seconds was measured for the MG problem.

8.13. The NEC Cenju-4

Machine type: RISC-based distributed-memory multi-processor.
Models: Cenju-4.
Operating system: Cenjuiox (Mach micro-kernel based Unix flavor).
Connection structure: Multistage crossbar.
Compilers: Fortran 77, Fortran 90, HPF (subset), ANSI C.
Vendors information Web page:
`kiefer.gmd.de:8002/popcorn/services/Overview.html`.
Year of introduction: 1998.

System parameters

Model	Cenju-4
Clock cycle	5 ns
Theor. peak performance	
Per Proc. (64-bits)	400 Mflop/s
Maximal	410 Gflop/s
Memory/node	≤512 MB
Memory/maximal	≤512 GB
No. of processors	8–1024
Communication bandwidth	
Point-to-point	200 MB/s

Remarks: The name Cenju-4 suggests that there have been predecessors, Cenju-1, Cenju-2, and Cenju-3. This is indeed the case but the first two systems have only been used internally by NEC for research purposes and were never officially marketed. The Cenju-3 was also placed externally but, again, mostly for evaluation purposes. The same is the case for the present Cenju-4: it is not actively marketed, although NEC will have no objections to selling it. Officially, the Cenju-series is regarded by NEC as systems to gain experience in massively parallel computing and to develop the proper tools for it.

The Cenju-4 is based on the MIPS R10000 RISC processor. All processors have, apart from their on-chip 32 KB primary data and instruction cache, a secondary cache of 1 MB to mitigate the problems that arise in the high data usage of the CPU.

The interconnection type used in the Cenju is a multistage crossbar build from 4×4 modules that are pipelined. So, in a full configuration the maximal number of levels in the crossbar to be traversed is six. The peak transfer rate of the crossbar is quoted as 200 MB/s irrespective of the data placement. Preliminary measurements of the author of this report show that the practical transfer rate for point-to-point communication is at least 175 MB/s with MPI; a quite high efficiency.

The system needs a front-end processor like the NEC EWS4800 (functionally equivalent to Silicon Graphics workstations) or SUN. The I/O requirements have to be fulfilled by the front-end system as the Cenju does not have local (distributed) I/O capabilities.

There is some software support that should make the programmer's life somewhat easier. The library PARALIB/CJ contains proprietary functions for forking processes, barrier synchronization, remote procedure calls, and block transfer of data. Like on the Cray T3E (8.17), the Hitachi SR8000 (8.9), and on the Meiko CS-2 (8.12) the programmer has the possibility to write/read directly to/from non-local memories which avoids much message passing overhead.

Measured Performances: No systematic performance measurement have been done yet on the Cenju-4. However, from comparative studies it seems that the speed on some applications is presently about 2/3 of an equivalent SGI R10000 node due to a different compiler technology (Cassirer and Steckel 1998). Nagel reports a speed of 90-100 Mflop/s for in-cache matrix-matrix multiplication in Fortran 90 per node (Nagel 1998).

8.14. The NEC SX-5 series

Machine type: Shared-memory multi-vector-processor.
Models: SX-5/8 SX-5/16, SX-5M.
Operating system: Super-UX (Unix variant based on BSD V.4.3 Unix).
Compilers: Fortran 90, HPF, ANSI C, C++.
Vendors information Web page: `www.ess.nec.de` → SX-5 Series.
Year of introduction: 1998.

System parameters

Model	SX-5/8	SX-5/16	SX-5Mx
Clock cycle	4 ns	4 ns	4 ns
Theor. peak performance			
Per Proc. (64-bits)	8 Gflop/s	8 Gflop/s	8 Gflop/s
Maximal			
Single frame:	64 Gflop/s	128 Gflop/s	—
Multi frame:	—	—	4 Tflop/s
Main Memory (per frame)	$\leq$64 GB	$\leq$128 GB	$\leq$4 TB
No. of processors	4–8	8–16	8–512

Remarks: The SX-5 series is offered is three models: a single-frame model that can house at most 8 CPUs (SX-5/8), a single-frame model containing up to 16 CPUs (SX-5/16), and multi-frame models (SX-5Mx) where $x = 2, \ldots, 32$ in which 2–32 single-frame systems are coupled into a larger system. There are two ways to couple the SX-5 frames in a multi-frame configuration: NEC provides a full crossbar, the so-called IXS crossbar to connect the various frames together at a speed of 16 GB/s for point-to-point out-of-frame communication (128 GB/s bi-sectional bandwidth for a maximum configuration). In addition, a HiPPI interface is available for inter-frame communication at lower cost and speed.

Every CPU contains 16 functional unit pipe sets. As the clock cycle is 4 ns and each pipe set is able to deliver 2 floating-point results per cycle, the total maximum performance is 8 Gflop/s per CPU. The bandwidth from memory to the CPUs is 32 64-bit words per cycle per CPU. With a replication factor of 16 pipe sets this is enough to provide two operands per pipe set but it is not sufficient to transport the results back to the memory at the same time. So, some trade-offs with the re-use of operands have to be made to attain the peak performance.

In contrast with its predecessor, the SX-4, the SX-5 is not offered anymore with faster, but more expensive and bulkier SRAM memory. The systems are exclusively manufactured with Synchronous DRAM memory.

The technology used is CMOS. This lowers the fabrication costs and the power consumption appreciably (the same approach is being used in the Fujitsu VPP700 and the SGI/Cray SV1, see section 8.7 and 8.16) and all models are air cooled.

For distributed computing there is an HPF compiler and for message passing optimized MPI (MPI/SX) is available.

Measured Performances: The first systems just have been delivered (Dec. 1998). Therefore, no independent performance figures are available at this time.

8.15. The Silicon Graphics Origin series

Machine type: RISC-based CC-NUMA system.
Models: Origin 200, Origin 2000.
Operating system: IRIX (SGI's Unix variant).
Connection structure: Crossbar, hypercube (see remarks).
Compilers: Fortran 77, Fortran 90, HPF, C, C++, Pascal.
Vendors information Web page:
`www.sgi.com/Products/hardware/servers/index.html`.
Year of introduction: 1996.

System parameters

Model	Origin 200	Origin 2000
Clock cycle	4 ns	4 ns
Theor. peak performance		
Per Proc. (64-bits)	500 Mflop/s	500 Mflop/s
Maximal	2 Gflop/s	64 Gflop/s
Memory	$\leq$4 GB	$\leq$256 GB
No. of processors	1–4	2–128
Communication bandwidth		
Point-to-point	780 MB/s	780 MB/s
Aggregate peak	3.1 GB/s	99.8 GB/s
Bisectional	1.6 GB/s	82 GB/s

Remarks: The Origin 2000 is the latest high-end parallel server marketed by SGI. The basic processor is the MIPS R10000 which is now offered at a clock cycle of 4 ns. A maximum of 128 processors can be configured in the system. The interconnection is somewhat hybrid: 4 CPUs on two node cards can communicate directly with the memory partitions of each other via the hub, a 4-ported non-blocking crossbar. Hubs can be coupled to other hubs in a hypercube fashion. Although the standard maximal configuration of an Origin2000 contains 128 processors, on special request larger systems can be build with the same technology. This was for instance done for the ASCI Blue Mountain project (ASCI 2001).

The machine is a typical representative of the CC-NUMA class of systems. The memory is physically distributed over the node boards but there is one system image. Because of the structure of the system, the bi-sectional bandwidth of the system remains constant from 4 processors on: 82 GB/s. This is a large improvement over the earlier PowerChallenge systems which possessed a 1.2 GB/s bus.

The Origin 200 is a smaller configuration, using the same crossbar as the Origin 2000 but without the need for the hypercube connections used in the latter. Effectively, it is a SMP system because of the uniform access of the memory modules. Therefore, also the bi-sectional bandwidth is identical to the point-to-point bandwidth: 1.6 GB/s.

Parallelization is done either automatically by the (Fortran or C) compiler or explicitly by the user, mainly through the use of directives. All synchronization, etc., has to be done via memory. This may cause potentially a fairly large parallelization overhead. Also a message passing model is allowed on the Origin using the optimized SGI versions of PVM, MPI, and the SGI/Cray-specific `shmem` library. Programs implemented in this way will possibly run very efficiently on the system.

A nice feature of the Origins is that it may migrate processes to nodes that should satisfy the data requests of these processes. So, the overhead involved in transferring data across the machine are minimized in this way. The technique is reminiscent of the late Kendall Square Systems although in these systems the data were moved to the active process. SGI claims that the time for non-local memory references is on average about 3 times longer than for local memory references.

Measured Performances: In Dongarra (1999) a speed of 1.608 Tflop/s out of 2.52 was measured on a system with 5040 processors on the ASCI Blue Mountain machine for the solution of a linear system with a size of 374400, an efficiency of 64%. Furthermore, an extensive benchmark report from the EuroBen Foundation is available for the regular Origin2000 configuration (van der Steen 1998).

8.16. The SGI/Cray Research Inc. SV1

Machine type: Shared-memory multi-vector-processor.
Models: SGI/Cray SV1-A, SV1-1, SV1 Supercluster.
Operating system: UNICOS (Cray Unix variant).
Compilers: Fortran 90, C, C++, Pascal, ADA.
Vendors information Web page: `www.sgi.com/sv1/`
Year of introduction: 1998.

System parameters

Model	SGI/Cray SV1-A	SGI/Cray SV1-1	SV1 Supercluster
Clock cycle	3.33 ns	3.33 ns	3.33 ns
Theor. peak perform.			
Per Proc.	1.2/4.8 Gflop/s	1.2/4.8 Gflop/s	1.2/4.8 Gflop/s
Maximal	19.2 Gflop/s	38.4 Gflop/s	1.2 Tflop/s
Memory	≤16 GB	≤32 GB	≤1 TB
No. of processors	8–16	8–32	≤1024
Memory bandwidth			
Memory–Cache	7.7 GB/s	7.7 GB/s	7.7 GB/s
Cache–CPU	9.6 GB/s	9.6 GB/s	9.6 GB/s
Aggregate	30.7 GB/s	61.4 GB/s	—
Node–node (bi-direct.)	—	—	1 GB/s

Remarks: The SGI/Cray SV1 is the successor both to the CMOS-based Cray J90 and the Cray T90 which was based on ECL technology. The SV1 systems are CMOS-based and therefore much cheaper to manufacture than the ECL-based systems. In this respect SGI is following the trend set in by Fujitsu and NEC a few years ago with their vector systems (see 8.7 and 8.14).

The single-cabinet configurations come in two sizes, the SV1-A and the SV1-1 that can house 4 and 8 processor boards, respectively. Each processor board contains 4 CPUs that can deliver a peak rate of 4 floating-point operations per cycle, amounting to a theoretical peak performance of 1.2 Gflop/s per CPU. However, 4 CPUs can be coupled *across* CPU boards in a configuration to form a so-called Multi Streaming Processor (MSP) resulting in a processing unit that has effectively a theoretical peak performance of 4.8 Gflop/s. The configuration into MSPs and/or single CPU combinations can be done via software at start-up time. The vector start-up time for the single CPUs is smaller than for MSPs, so for small vectors single CPUs might be preferable while for programs containing long vectors the MSPs should be of advantage. The number of combinations that can be made is large but at least 8 CPUs must be configured as single 1.2 Gflop/s CPUs. So a full SV1-1 cabinet may be configured as 32 single 1.2 Gflop/s CPUs or as 1–6 MSPs with the remaining processors as single CPUs.

Another new feature in the SV1 is a combined scalar and vector cache of 256 KB per CPU. This cache is important because the bandwidth of 7.7 GB/s per CPU board amounts to only 0.8 eight-byte operands per cycle. The cache can ship 1 operand per cycle to a CPU. This relative bandwidth is much smaller than what was offered in the former Cray systems which makes the cache all the more important.

Like in the NEC SX-5 single cabinets can be combined to form a cluster (Supercluster in SGI/Cray terminology) by a so-called GigaRing. The GigaRing, which is also used to coupled I/O sub-systems, is comprised of two counter-rotating rings with a bandwidth of 1 GB/s each. Where the systems in a cabinet are SM-MIMD systems, a multi-cabinet Supercluster is an DM-MIMD system and can be operated in parallel only by some parallel programming model like MPI or HPF.

Measured Performances: The importance of the cache is well illustrated by the performance of a matrix-matrix multiplication as occurs in the EuroBen Benchmark. With a single processor (called Single Stream Processing, SSP, by SGI) and with the cache a peak speed of 999 Mflop/s is observed at a matrix order of $n = 300$ and decreasing to a speed of 666 Mflop/s at an order of $n = 1000$. Without the cache the speed reaches at an order of $n = 300$ a speed of 625 Mflop/s and slowly increases to about 650 Mflop/s at $n = 1000$. In MSP mode this behavior is similar: with cache the speed at $n = 100$ is 2.61 Gflop/s, decreasing to 1.41 Gflop/s at $n = 1000$. Without the cache the observed speed at $n = 100$ is 1.0 Gflop/s and rises to 1.4 Gflop/s at $n = 1000$. This means that for modestly sized problems the cache can boost the performance with a factor 1.5–2. The efficiency in MSP mode is presently not too high: just over 50% in a favorable situation. As the MSP facility is very new, one may expect that the efficiency will increase considerably in the near future.

8.17. The SGI/Cray Research Inc. T3E

Machine type: RISC-based distributed-memory multi-processor.
Models: T3E, T3E-900, T3E-1200.
Operating system: UNICOS/mk (micro kernel-based Unix).
Connection structure: 3-D Torus.
Compilers: Fortran 77, Fortran 90, HPF, ANSI C, C++.
Vendors information Web page:
`www.cray.com/products/systems/crayt3e/`

Year of introduction: T3E, T3E-900: 1996, T3E-1200: 1997.

System parameters

Model	T3E-900	T3E-1200
Clock cycle	2.2 ns	1.67 ns
Theor. peak performance		
Per Proc. (64-bits)	900 Mflop/s	1200 Mflop/s
Maximal	1843 Gflop/s	2458 Gflop/s
Memory/node	$\leq$ 2 GB	$\leq$ 2 GB
Memory/maximal	$\leq$ 4 TB	$\leq$ 4 TB
No. of processors	6–2048	6–2048
Communication bandwidth		
Point-to-point	300 MB/s	300 MB/s

Remarks: The T3E is the second generation of DM-MIMD systems from SGI/Cray. Lexically, it follows in name after its predecessor T3D which name referred to its connection structure: a 3-D torus. In this respect it has still the same interconnection structure as the T3D. In many other respects, however, there are quite some differences. A first and important difference is that no front-end system is required anymore (although it is still possible to connect to SGI/Cray vector systems). The systems up to 128 processors are air-cooled. The larger ones, from 256-2048 processors, are liquid cooled.

The T3E uses the DEC Alpha 21164 for its computational tasks just like the Avalon A12 (see 8.2). In 1997, a T3E-1200 was introduced that uses 21164A processors at a clock rate of only 1.67 ns but that is identical in any other aspect to the T3E-900. SGI stresses, that the processors are encapsulated in such a way that they can be exchanged easily for any other (faster) processor as soon as this would be available without affecting the macro-architecture of the system.

Each node in the system contains one processing element (PE) which in turn contains a CPU, memory, and a communication engine that takes care of communication between PEs. The bandwidth between nodes is quite high: 300 MB/s, bi-directional. Like the T3D, the T3E has hardware support for fast synchronization. E.g., barrier synchronization takes only one cycle per check.

In the micro-architecture most changes have taken place with the transition from the T3D to the T3E. Firstly, there is only one CPU per node instead of two, which removes a source of asymmetry between processors. Secondly, the new node processor has a 96 KB 3-way set-associative secondary cache which may relieve some of the problems of data fetching that were present in the T3D where only a primary cache was present. Third, the Block Transfer Engine has been replaced by a set of E-registers and streaming registers that are much more flexible and that remove some odd restrictions on the size of shared arrays and the number processes when using Cray-specific PVM. An interesting additional feature is the availability of 32 contexts per processor which opens the door for multiprocessing.

In the T3D all I/O had to be handled by the front-end, a system at least from the Cray Y-MP/E class. In the T3E distributed I/O is present. For every 8 PEs an I/O channel can be configured in the air-cooled systems and 1 I/O channel per 16 nodes in the liquid-cooled systems. The maximum bandwidth for a channel is about 1 GB/s, the actual speed will be in the order of 700 MB/s.

The T3E supports various programming models. Apart from PVM3 and MPI for message passing and HPF for data distribution, a Cray proprietary one-sided communication library, the so-called `shmem` library can be employed for message passing. In addition, the BSP library (see Hill et al. (1997)), also a one-sided message passing library is available. The `shmem` library is implemented close to the hardware and shows a very low latency of only 1.6 μs.

Measured Performances: In Dongarra (1999) a speed of 1.127 Tflop/s is reported for the solution of a dense linear system of order 148800 on a T3E-1200 with 1488 processors. The efficiency for such an exercise is 63%.

8.18. The Sun E10000 Starfire

Machine type: RISC-based shared-memory multiprocessor.
Models: E10000 Starfire.
Operating system: Solaris (Sun's Unix flavor).
Compilers: Fortran 77, Fortran 90, C, C++.
Vendors information Web page:
`www.sun.com/servers/ultra_enterprise/10000/`
Year of introduction: 1997.

System parameters

Model	E10000
Clock cycle	4 ns
Theor. peak performance	
Per Proc. (64-bits)	500 Mflop/s
Maximal	32 Gflop/s
Main Memory	$\leq$ 64 GB
No. of processors	16–64

Remarks: The Starfire E10000 is the largest of a series of Ex000 servers, where x can be 3, 4, 5, 6, 10. We only discuss this largest model as Sun has clearly positioned this machine themselves as a system for large-scale high performance computing. The basic processor is a 4 ns cycle UltraSPARC processor with a theoretical peak performance of 500 Mflop/s. Up to 64 processors are connected by a 64×64 crossbar, the largest crossbar employed commercially. This crossbar, called the Gigaplane-XB, also makes it different from the lower-end models from the Ex000 series as these systems use a bus interconnect between processors. The system is built up from system boards each containing up to 4 processors, 2 level-2 caches ($\leq$ 4 MB) and 4 memory banks that plug into the Gigaplane crossbar which thus acts as a backplane. The caches are kept coherent by a "snoopy bus" protocol, i.e., each cache is aware of the (in)validation of data by continuous monitoring the data on the backplane and updating their copies accordingly.

The Gigaplane crossbar connects to the processors with separate data and address lines which recognizes the fact that most data transfers are essentially point-to-point transfers while addresses often have to be broadcasted to many or all processors. The effective aggregate bandwidth for data is 102.4 GB/s with a point-to-point speed of 1.6 GB/s (theoretical peak).

The Starfire is a typical SMP machine with provisions for shared-memory parallelism in the Fortran and C(++) compilers by directives in the source code. Sun has joined the OpenMP consortium for standardizing the shared-memory programming model. Of course it is possible to cluster E10000s servers and use such a cluster in a DM-MIMD way.
Measured Performances: In Dongarra (1999) a speed of 26.45 Gflop/s is reported for a 64 processor machine in solving an order 19968 linear system. The efficiency for this problem is 83%. In Dongarra (1999) also results for a 4-way cluster with a clock cycle of 3 ns are reported. This 256 processor system reached a speed of 123.9 Gflop/s in solving a linear system of order 80640. This amounts to an efficiency of 72%.

8.19. The Tera MTA

Machine type: Distributed-memory multi-processor.
Models: MTA-xC, $x = 1, \ldots, 256$.
Operating system: Unix BSD4.4 + proprietary micro kernel.
Compilers: Fortran 77 (Fortran 90 extensions), HPF, ANSI C, C++.
Vendors information Web page: `www.tera.com/`
Year of introduction: 1997.

System parameters

Model	MTA-xC
Clock cycle	3 ns
Theor. peak performance	
Per Proc. (64-bits)	1 Gflop/s
Maximal	256 Gflop/s
Main Memory	$\leq$ 256 GB
No. of processors	16–256

Remarks: Although the memory in the MTA is physically distributed, the system is emphatically presented as a shared-memory machine (with non-uniform access time). The latency incurred in memory references is hidden by *multi-threading*, i.e., usually many concurrent program threads (instruction streams) may be active at any time. Therefore, when for instance a load instruction cannot be satisfied because of memory latency the thread requesting this operation is stalled and another thread of which an operation can be done is switched into execution. This switching between program threads only takes 1 cycle. As there may be up to 128 instruction streams per processor and 8 memory references can be issued without waiting for preceding ones, a latency of 1024 cycles can be tolerated. References that are stalled are retried from a retry pool. A construction that worked out similarly was to be found in the late Stern Computing Systems SSP machines (see in section 9).

The connection network connects a 3-D cube of p processors with sides of $p^{1/3}$ of which alternately the x- or y axes are connected. Therefore, all nodes connect to four out of six neighbors. In a p processor system the worst case latency is $4.5p^{1/3}$ cycles; the average latency is $2.25p^{1/3}$ cycles. Furthermore, there is an I/O port at every node. Each network port is capable of sending and receiving a 64-bit word per cycle which amounts to a bandwidth of 5.33 GB/s per port. In case of detected failures, ports in the network can be bypassed without interrupting operations of the system.

Although the MTA should be able to run "dusty-deck" Fortran programs because parallelism is automatically exploited as soon as an opportunity is detected for multi-threading, it may be (and often is) worthwhile to explicitly control the parallelism in the program and to take advantage of known data locality occurrences. MTA provides handles for this in the form of compiler directives, library routines, including synchronization, barrier, and reduction operations on defined groups of threads. Controlled and uncontrolled parallelism approaches may be freely mixed. Furthermore, each variable has a full/empty bit associated with it which can be used to control parallelism and synchronization

with almost zero overhead. HPF will also be supported for SPMD-style programming.
Measured Performances: The company has presently delivered a 4-processor system to the San Diego Supercomputing Center. This system runs at a clock cycle of 4.4 ns instead of the planned 3 ns. Consequently, the peak performance of a processor is 450 Mflop/s. Using the EuroBen Benchmark (van der Steen 1991) a performance of 388 Mflop/s out of 450 Mflop/s was found for an order 800 matrix-vector multiplication, an efficiency of 86%. On 4 processors a speed of 1 Gflop/s out of 1.8 Gflop/s was found, an efficiency of 56% on the same problem. For 1-D FFTs up to 1 million elements a speed of 106 Mflop/s was found on 1 processor and the about the same speed on 4 processors due to an insufficient availability of parallel threads.

9. Systems that disappeared from the list

As already stated in the introduction the list of systems is not complete. On one hand this is caused by the sheer number of systems that are presented to the market and are often very similar to systems described above (for instance, the Volvox system not listed was very similar but not equivalent to the former C-DAC system and there are numerous other examples). On the other hand, there are many systems that are still in operation around the world, often in considerable quantities that for other reasons are excluded. The most important reasons are:

- The system is not marketed anymore. This is generally for one of two reasons:
    - The manufacturer is out of business.
    - The manufacturer has replaced the system by a newer model of the same type or even of a different type.
- The system has become technologically obsolete in comparison to others of the same type. Therefore, listing them is not sensible anymore.

Below we present a table of systems that fall into one of the categories mentioned above. We think this may have some sense to those who come across machines that are still around but are not the latest in their fields. It may be interesting at least to have an indication how such systems compare to the newest ones and to place them in context.

It is good to realism that although systems have disappeared from the section above they still may exist and are actually sold. However, their removal stems in such cases mainly from the fact that they are not serious candidates for high-performance computing anymore.

The table is, again, not complete and admittedly somewhat arbitrary. The data are in a highly condensed form: the system name, system type, theoretical maximum performance of a fully configured system, and the reason for their disappearance is given. The arbitrariness lies partly in the decision which systems are still sufficiently of interest to include and which are not.

We include also both the year of introduction and the year of exit of the systems when they were readily accessible. These time-spans could give a hint of the dynamics that governs this very dynamical branch of the the computer industry.

- **Machine**: The Alex AVX 2.
  - **Year of introduction**: 1992.
  - **Year of exit**: 1997.
  - **Type**: RISC-based distributed-memory multi-processor.
  - **Theoretical Peak performance** : 3.84 Gflop/s.
  - **Reason for disappearance**: System is obsolete, there is no new system planned.
- **Machine**: Alliant FX/2800.
  - **Year of introduction**: 1989.
  - **Year of exit**: 1992.
  - **Type**: Shared-memory vector-parallel, max. 28 processors.
  - **Theoretical Peak performance**: 1120 Mflop/s.
  - **Reason for disappearance**: Manufacturer out of business.
- **Machine**: The AxilSCC.
  - **Year of introduction**: 1996.
  - **Year of exit**: 1997.
  - **Type**: RISC-based distributed-memory system, max. 512 processors.
  - **Theoretical Peak performance** : 76.8 Gflop/s.
  - **Reason for disappearance**: System is not marketed anymore by Axil.
- **Machine**: BBN TC2000.
  - **Year of introduction**: ?

- **Year of exit**: 1990.
- **Type**: Virtual shared-memory parallel, max. 512 processors.
- **Theoretical Peak performance**: 1 Gflop/s.
- **Reason for disappearance**: Manufacturer has discontinued marketing parallel computer systems.

■ **Machine**: Cambridge Parallel Processing DAP Gamma.

- **Year of introduction**: 1986.
- **Year of exit**: 1995.
- **Type**: Distributed-memory processor array system, max. 4096
- processors. **Theoretical Peak performance**: 1.6 Gflop/s (32-bit).
- **Reason for disappearance**: replaced by newer Gamma II Plus series (8.3).

■ **Machine**: C-DAC PARAM 9000/SS.

- **Year of introduction**: 1995.
- **Year of exit**: 1997.
- **Type**: Distributed-memory RISC based system, max. 200 processors.
- **Theoretical Peak performance**: 12.0 Gflop/s.
- **Reason for disappearance**: replaced by newer PARAM OpenFrame series (8.4).

■ **Machine**: Convex SPP-1000/1200/1600.

- **Year of introduction**: 1995 (SPP-1000).
- **Year of exit**: 1996 (SPP-1600).
- **Type**: Distributed-memory RISC based system, max. 128 processors.
- **Theoretical Peak performance**: 25.6 Gflop/s
- **Reason for disappearance**: replaced by newer HP Exemplar V2500 system (8.10).

■ **Machine**: Cray Computer Corporation Cray-2.

- **Year of introduction**: 1982.
- **Year of exit**: 1992.

  - **Type**: Shared-memory vector-parallel, max. 4 processors.
  - **Theoretical Peak performance**: 1.95 Gflop/s.
  - **Reason for disappearance**: Manufacturer out of business.

- **Machine**: Cray Computer Corporation Cray-3.
  - **Year of introduction**: 1993.
  - **Year of exit**: 1996.
  - **Type**: Shared-memory vector-parallel, max. 16 processors.
  - **Theoretical Peak performance**: 16 Gflop/s.
  - **Reason for disappearance**: Manufacturer out of business.

- **Machine**: Cray Research Inc. APP.
  - **Year of introduction**: 1993.
  - **Year of exit**: 1995.
  - **Type**: Shared-memory RISC based system, max. 84 processors.
  - **Theoretical Peak performance**: 6.7 Gflop/s.
  - **Reason for disappearance**: Product line discontinued, gap was filled by Cray J90 (see below).

- **Machine**: Cray T3D.
  - **Year of introduction**: 1994.
  - **Year of exit**: 1996.
  - **Type**: Distributed-memory RISC based system, max. 2048 processors.
  - **Theoretical Peak performance**: 307 Gflop/s.
  - **Reason for disappearance**: replaced by newer Cray T3E (8.17).

- **Machine**: Cray T3E Classic.
  - **Year of introduction**: 1996.
  - **Year of exit**: 1997.
  - **Type**: Distributed-memory RISC based system, max. 2048 processors.
  - **Theoretical Peak performance**: 1228 Gflop/s.

  - **Reason for disappearance**: replaced Cray T3Es with faster clock. (8.17).

- **Machine**: Cray J90.
  - **Year of introduction**: 1994.
  - **Year of exit**: 1998.
  - **Type**: Shared-memory vector-parallel, max. 32 processors.
  - **Theoretical Peak performance**: 6.4 Gflop/s.
  - **Reason for disappearance**: replaced by newer SGI/Cray SV1 (8.16).

- **Machine**: Cray Research Inc. Cray Y-MP, Cray Y-MP M90.
  - **Year of introduction**: 1989 (Cray Y-MP).
  - **Year of exit**: 1994 (Cray Y-MP M90).
  - **Type**: Shared-memory vector-parallel, max. 8 processors.
  - **Theoretical Peak performance**: 2.6 Gflop/s.
  - **Reason for disappearance**: replaced by newer C90 (see below).

- **Machine**: Cray Y-MP C90.
  - **Year of introduction**: 1994.
  - **Year of exit**: 1996.
  - **Type**: Shared-memory vector-parallel, max. 16 processors.
  - **Theoretical Peak performance**: 16 Gflop/s.
  - **Reason for disappearance**: replaced by newer T90 (see below).

- **Machine**: Cray Y-MP T90.
  - **Year of introduction**: 1995.
  - **Year of exit**: 1998.
  - **Type**: Shared-memory vector-parallel, max. 32 processors.
  - **Theoretical Peak performance**: 58 Gflop/s.
  - **Reason for disappearance**: replaced by newer SGI/Cray SV1 (8.16).

- **Machine**: Digital Equipment Corp. Alpha farm.
  - **Year of introduction**: —.

  - **Year of exit**: 1994.
  - **Type**: Distributed-memory RISC based system, max. 4 processors.
  - **Theoretical Peak performance**: 0.8 Gflop/s.
  - **Reason for disappearance**: replaced by newer AlphaServer clusters (see below).

- **Machine**: Digital Equipment Corp. AlphaServer 8200 & 8400.
  - **Year of introduction**: —.
  - **Year of exit**: 1998.
  - **Type**: Distributed-memory RISC based systems, max. 6 processors
  - (AlphaServer 8200) or 14 (AlphaServer 8400). **Theoretical Peak performance**: 7.3 Gflop/s, resp. 17.2 Gflop/s.
  - **Reason for disappearance**: replaced by newer Compaq GS60 and GS140 (8.5).

- **Machine**: Fujitsu AP1000.
  - **Year of introduction**: 1991.
  - **Year of exit**: 1996.
  - **Type**: Distributed memory RISC based system, max. 1024 processors.
  - **Theoretical Peak performance**: 5 Gflop/s.
  - **Reason for disappearance**: replaced by the AP3000 systems (8.6).

- **Machine**: Fujitsu VPP500 series.
  - **Year of introduction**: 1993.
  - **Year of exit**: 1995.
  - **Type**: Distributed-memory multi-processor vector-processors, max.
  - 222 processors. **Theoretical Peak performance**: 355 Gflop/s.
  - **Reason for disappearance**: replaced by the VPP300/700 series (8.7).

- **Machine**: Fujitsu VPX200 series.
  - **Year of introduction**: —.

- **Year of exit**: 1995.
- **Type**: Single-processor vector-processors.
- **Theoretical Peak performance**: 5 Gflop/s.
- **Reason for disappearance**: replaced by the VPP300/700 series (8.7).

■ **Machine**: Hitachi S-3800 series.

- **Year of introduction**: 1993.
- **Year of exit**: 1998.
- **Type**: Shared-memory multi-processor vector-processors, max. 4
- processors. **Theoretical Peak performance**: 32 Gflop/s.
- **Reason for disappearance**: Replaced by the SR8000 (8.9).

■ **Machine**: Hitachi SR2001 series.

- **Year of introduction**: 1994.
- **Year of exit**: 1996.
- **Type**: Distributed-memory RISC based system, max. 128 processors.
- **Theoretical Peak performance**: 23 Gflop/s.
- **Reason for disappearance**: Replaced by successor SR2201 (see below).

■ **Machine**: Hitachi SR2201 series.

- **Year of introduction**: 1996.
- **Year of exit**: 1998.
- **Type**: Distributed-memory RISC based system, max. 1024 processors.
- **Theoretical Peak performance**: 307 Gflop/s.
- **Reason for disappearance**: Replaced by the newer SR8000 (8.9).

■ **Machine**: HP/Convex C4600 series.

- **Year of introduction**: 1994.
- **Year of exit**: 1997.
- **Type**: Shared-memory vector-parallel, max. 4 processors.

  - **Theoretical Peak performance**: 3.24 Gflop/s.
  - **Reason for disappearance**: HP does not market the vector product line anymore.

- **Machine**: IBM ES/9000 series.
  - **Year of introduction**: 1991.
  - **Year of exit**: 1994.
  - **Type**: Shared-memory vector-parallel system, max. 6 processors.
  - **Theoretical Peak performance**: 2.67 Gflop/s.
  - **Reason for disappearance**: IBM does not pursue high-performance computing by this product line anymore.

- **Machine**: IBM SP1 series.
  - **Year of introduction**: 1992.
  - **Year of exit**: 1994.
  - **Type**: Distributed-memory RISC based system, max. 64 processors.
  - **Theoretical Peak performance**: 8 Gflop/s.
  - **Reason for disappearance**: Replaced by the newer RS/6000 SP (8.11).

- **Machine**: Intel Paragon XP.
  - **Year of introduction**: 1992.
  - **Year of exit**: 1996.
  - **Type**: Distributed-memory RISC based system, max. 4000 processors.
  - **Theoretical Peak performance**: 300 Gflop/s.
  - **Reason for disappearance**: Except for a non-commercial research system (the ASCI Option Red system at Sandia National Labs.) Intel is not in the business of high-performance computing anymore.

- **Machine**: Kendall Square Research KSR2.
  - **Year of introduction**: 1992.
  - **Year of exit**: 1994.

- **Type**: Virtually shared-memory parallel, max. 1088 processors.
- **Theoretical Peak performance**: 400 Gflop/s.
- **Reason for disappearance**: Kendall Square has terminated its business.

- **Machine**: Kongsberg Informasjonskontroll SCALI.
  - **Year of introduction**: 1996.
  - **Year of exit**: 1997.
  - **Type**: Distributed-memory RISC based system, max. 512 processors.
  - **Theoretical Peak performance**: 76.8 Gflop/s.
  - **Reason for disappearance**: Kongsberg does not market the system anymore.
- **Machine**: MasPar MP-1, MP-2.
  - **Year of introduction**: 1991 (MP-1).
  - **Year of exit**: 1996.
  - **Type**: Distributed-memory processor array system, max. 16384
  - processors. **Theoretical Peak performance**: 2.4 Gflop/s (64-bit, MP-2).
  - **Reason for disappearance**: Systems are not marketed anymore.
- **Machine**: Matsushita ADENART.
  - **Year of introduction**: 1991.
  - **Year of exit**: 1997.
  - **Type**: Distributed-memory RISC based system, 256 processors.
  - **Theoretical Peak performance**: 2.56 Gflop/s.
  - **Reason for disappearance**: Machine is obsolete and no new systems are developed in this line.
- **Machine**: Meiko CS-1 series.
  - **Year of introduction**: 1989.
  - **Year of exit**: 1995.
  - **Type**: Distributed-memory RISC based system.

  - **Theoretical Peak performance**: 80 Mflop/s per processor.
  - **Reason for disappearance**: Replaced by the newer CS-2 (8.12).

- **Machine**: nCUBE 2S.
  - **Year of introduction**: 1993.
  - **Year of exit**: 1998.
  - **Type**: Distributed-memory system, max. 8192 processors.
  - **Theoretical Peak performance**: 19.7 Gflop/s.
  - **Reason for disappearance**: NCUBE has withdrawn from the scientific and technical market. The nCUBE 2S is now offered as a parallel multimedia server.

- **Machine**: nCUBE 3.
  - **Year of introduction**: — .
  - **Year of exit**: —
  - **Type**: Distributed-memory system, max. 10244 processors.
  - **Theoretical Peak performance**: 1 Tflop/s.
  - **Reason for disappearance**: Was announced several times as the successor of the nCUBE 2S (see above) but was never realized.

- **Machine**: NEC Cenju-3.
  - **Year of introduction**: 1994.
  - **Year of exit**: 1996.
  - **Type**: Distributed-memory system, max. 256 processors.
  - **Theoretical Peak performance**: 12.8 Gflop/s.
  - **Reason for disappearance**: replaced by newer Cenju-4 series (8.13).

- **Machine**: NEC SX-3R.
  - **Year of introduction**: 1993.
  - **Year of exit**: 1996.
  - **Type**: Shared-memory multi-processor vector processors, max. 4
  - processors.

    Theoretical Peak performance: 25.6 Gflop/s.

  - **Reason for disappearance**: replaced by newer SX-4 series (see below).

- **Machine**: NEC SX-4.
  - **Year of introduction**: 1995.
  - **Year of exit**: 1996.
  - **Type**: Distributed-memory cluster of SM-MIMD vector processors,
  - max. 256 processors. **Theoretical Peak performance**: 1 Tflop/s.
  - **Reason for disappearance**: replaced by newer SX-5 series (8.14).

- **Machine**: Parsys SN9000 series.
  - **Year of introduction**: 1993.
  - **Year of exit**: 1995.
  - **Type**: Distributed-memory RISC based system, max. 2048.
  - **Theoretical Peak performance**: 51.2 Gflop/s.
  - **Reason for disappearance**: Replaced by the newer TA9000 (but see below).

- **Machine**: Parsys TA9000 series.
  - **Year of introduction**: 1995.
  - **Year of exit**: 1996.
  - **Type**: Distributed-memory RISC based system, max. 512 processors.
  - **Theoretical Peak performance**: 119.3 Gflop/s.
  - **Reason for disappearance**: Parsys does not offer complete system anymore. Instead it sells node cards based on the TA9000 for embedded systems.

- **Machine**: Parsytec GC/Power Plus.
  - **Year of introduction**: 1993.
  - **Year of exit**: 1996.
  - **Type**: Distributed-memory RISC based system.
  - **Theoretical Peak performance**: 266.6 Mflop/s per processor.

  - **Reason for disappearance**: System has been replaced by the Parsytec CC systems (see below).

- **Machine**: Parsytec CC series.
  - **Year of introduction**: 1996.
  - **Year of exit**: 1998.
  - **Type**: Distributed-memory RISC based system.
  - **Theoretical Peak performance**: unspecified.
  - **Reason for disappearance**: Vendor has withdrawn from the High-Performance computing market.

- **Machine**: Siemens-Nixdorf VP2600 series.
  - **Year of introduction**: ?
  - **Year of exit**: 1995.
  - **Type**: Single-processor vector-processors.
  - **Theoretical Peak performance**: 5 Gflop/s.
  - **Reason for disappearance**: replaced by the VPP300/700 series (8.7).

- **Machine**: Silicon Graphics PowerChallenge.
  - **Year of introduction**: 1994.
  - **Year of exit**: 1996.
  - **Type**: Shared-memory multi-processor, max. 36 processors.
  - **Theoretical Peak performance**: 14.4 Gflop/s.
  - **Reason for disappearance**: replaced by the SGI Origin 2000 (8.15).

- **Machine**: Stern Computing Systems SSP.
  - **Year of introduction**: 1994.
  - **Year of exit**: 1996.
  - **Type**: Shared-memory multi-processor, max. 6 processors.
  - **Theoretical Peak performance**: 2 Gflop/s.
  - **Reason for disappearance**: Vendor terminated its business just before delivering first systems.

- **Machine**: Thinking Machine Corporation CM-2(00).
  - **Year of introduction**: 1987.

  - **Year of exit**: 1991.
  - **Type**: SIMD parallel machine with hypercube structure, max. 64K
  - processors. **Theoretical Peak performance**: 31 Gflop/s.
  - **Reason for disappearance**: was replaced by the newer CM-5 (but see below).

- **Machine**: Thinking Machine Corporation CM-5.
  - **Year of introduction**: 1991.
  - **Year of exit**: 1996.
  - **Type**: Distributed-memory RISC based system, max. 16K processors.
  - **Theoretical Peak performance**: 2 Tflop/s.
  - **Reason for disappearance**: Thinking Machine Corporation has stopped manufacturing hardware and hopes to keep alive as a software vendor.

10. Systems under development

Although we mainly want to discuss real, marketable systems and not experimental, special purpose, or even speculative machines, we want to include a section on systems that are in a far stage of development and have a fair chance of reaching the market. For inclusion in section 3 we set the rule that the system described there should be on the market within a period of 6 months from announcement. The systems described in this section will in all probability appear within one year from the publication of this report. However, there are vendors who do not want to disclose any specific data on their new machines until they are actually beginning to ship them. We recognize the wishes of such vendors (it is generally wise not to stretch the expectation of potential customers too long) and will not disclose such information.

Below we discuss systems that may lead to commercial systems to be introduced on the market between somewhat more than half a year to a year from now. The commercial systems that result from it will sometimes deviate significantly from the original research models depending on the way the development is done (the approaches in Japan and the USA differ considerably in this respect) and the user group which is targeted.

The year 1998 has been fruitful in terms of new systems: 3 large vendors came out with new systems: Hitachi with the SR8000, NEC with the SX-5, and SGI/Cray with the SV1. However, new ASCI contracts have been made and other vendors are due to come up with new systems.

10.1. Compaq/DEC

The present Compaq/DEC GS60 and GS140 servers (8.5) are in fact relabeled AlphaServers 8200 and 8400. They are still bus-based systems that cannot be scaled up effectively. As Compaq/DEC has won an ASCI contract to build a 30 Tflop/s machine by 2001, it is clear that the structure of the system has to change. Although such plans are not officially released it seems certain that a multi-level crossbar based on the fast DEC Memory Channel will be used. For instance, using an 8×8 crossbar an aggregate bandwidth of 8 GB/s per crossbar level could be reached. Using the next generation EV7 Alpha chip at a 0.9 ns clock cycle an 8-way cluster would provide a 35.2 Gflop/s building block. The system will probably be presented as a CC-NUMA machine.

10.2. Fujitsu

Fujitsu is the only large vector-processor vendor that did not present a new system by the end of 1998. The introduction of the successor to the VPP700 is expected in the spring of this year. As usual Fujitsu has been extremely tight-lipped about the new systems and no details are known. As the VPP700 has been a reasonably successful machine it stands to reason that the structure of its successor will be (almost) the same: vector-processors connected by a multi-level crossbar. The speed of both components will have undoubtly increased. In contrast to NEC and SGI/Cray, Fujitsu did not present a shared-memory image for the first-level clusters in its VPP700, except by its proprietary extended Fortran and HPF. In the next generation system this omission will very probably be mended.

10.3. Hewlett-Packard

The HP Exemplar systems consist of 32-CPU cluster nodes that are connected by 1 GB/s SCI rings to increase the system size (8.10). Although the speed of the PA-RISC processors have increased much over the years, the bandwidth of the rings stayed the same. This is obviously becoming a bottleneck in the scalability of the HP systems. Therefore, in the near future a new system with a different structure may be expected. In keeping with the majority of systems that are built presently one may expect that also HP will decide to connect its SMP clusters by a multi-level crossbar. As the point-to-point bandwidth between clusters certainly not will decrease, e.g., a 16×16 crossbar would need at least an aggregate bandwidth of 16 GB/s. As the present Exemplars already are CC-NUMA systems, the next generation certainly will also be of this class. At the moment HP keeps the options open whether the next system will be based on the next generation of the PA-RISC system or that the new IA-64 chip will be used.

10.4. IBM

IBM was, together with Intel and SGI/Cray one of the first ASCI contractors and has as such built the ASCI Blue Pacific machine with a peak speed of more than 1 Tflop/s. This system is based on the POWER3 chip (see 8.11). In following contracts IBM ventures to make a 100 Tflop/s system based on the successor of the POWER3 chip and with a higher level of clustering. At this moment the ASCI Blue machine has four-way clustered nodes. This might stay the same in the near future. With an increase in the clock speed from 3.3 to 2.2 ns a single CPU with four floating-point units already would have a peak speed of 1.8 Gflop/s which would give an four-way node a performance of 7.2 Gflop/s. It is to be expected that the switching fabric also will be altered. The High-Performance switch, as it is, would clearly form a bottleneck for the scalability. So, as in most other new systems (and in the present RS/6000 SP) a multi-level crossbar is highly probable. A new system could for instance consist of packages of 4 four-way nodes of approximately 30 Gflop/s peak performance which in turn are four-way connected, etc.

10.5. SGI

After the introduction of the SGI/Cray SV1 a new RISC processor based machine may be expected to replace the successful Origin2000 system. As SGI has loosened its ties with MIPS, the current source of Origin CPUs, SGI might choose another chip to base the Origin successor on. As SGI is also developing on Intel platforms now, the Intel IA-64 might be a possible choice when no MIPS processors would be employed. Whatever is chosen, the nodes need a to have speed of 1 Gflop/s or greater to be competitive with similar systems available around the year 2000. In most other respects the system might have largely the structure of the Origin2000. According to the SGI road map, the T3E line will at least for the next generation not be fused with the Origin successor and so will have a distinct architecture. What is sure is that the new system will retain the successful CC-NUMA concept of the present machine.

For SGI the integration of the machine lines is of extreme importance. Only a few vendors have as much as 2 product lines that are marketed (Fujitsu, Hitachi, NEC) where it is not always clear what are the benefits of one product line over the other. SGI still maintains no less than three product lines in high-performance computing, each with its own virtues and customer base. It is a highly complicated balancing act to reduce the product lines to two or even one without disappointing a part of the customers.

Acknowledgments

It is not possible to thank all people that have been contributing to this overview. Many vendors and people interested in this project have been so kind to provide me with the vital information or to correct us when necessary. Therefore, we will have to thank them here collectively but not less heartily for their support.

Special thanks are due to Greg Astfalk of Hewlett-Packard for valuable comments on the architecture section and on the HP Exemplar in particular.

Bibliography

C. Amza, A.L. Cox, S. Dwarkadas, P. Keleher, Honghui Lu, R. Rajamony, Weimin Yu, and W. Zwaenepoel. TreadMarks: Shared Memory Computing on Networks of Workstations. *IEEE Computer*, 29:18–28, 1996.

ASCI. The asci program, 2001. `http://www.llnl.gov/asci/`.

K. Cassirer and B. Steckel. Block-Structured Multigrid on the Cenju. In *2nd Cenju Workshop*, 1998.

D.E. Culler, J.P. Singh, and A. Gupta. *Parallel Computer Architecture: A Hardware/Software Approach.* Morgan Kaufmann Publishers Inc., 1998.

J.J. Dongarra. Performance of various computers using standard linear equations software. Technical Report CS-89-85, Computer Science Department, Univ. of Tennessee, 1999.

P. Flanders. Matrix Multiplication on 'C' series DAPs, 1991. AMT Document TR40.

M.J. Flynn. Some computer organisations and their effectiveness. *IEEE Trans. on Comp.*, C-21:948–960, 1972.

High Performance Fortran Forum. High Performance Fortran Language Specification. *Scientific Programming*, 2:1–170, 1993.

A. Geist, A. Beguelin, J. Dongarra, R. Manchek, W. Jaing, and V. Sunderam. *PVM: A Users' Guide and Tutorial for Networked Parallel Computing.* MIT Press, 1994.

W. Gropp, S. Huss-Ledermann, A. Lumsdaine, E. Lusk, B. Nitzberg, W. Saphir, and M. Snir. *MPI: The Complete Reference, Vol. 2, The MPI Extensions.* MIT Press, 1998.

J.M.D. Hill, W. McColl, D.C. Stefanescu, M.W. Goudreau, K. Lang, S.B. Rao, T. Suel, T. Tsantilas, and R. Bisseling. BSPlib: The BSP Programming Library. Technical Report PRG-TR-29-9, Oxford University Computing Laboratory, 1997.

2001. `www.hitachi.co.jp/Prod/comp/hpc/eng/sr1.html`.

R. W. Hockney and C. R. Jesshope. *Parallel Computers II.* Adam Hilger, 1987.

T. Horie, H. Ishihata, T. Shimizu, S. Kato, S. Inano, and M. Ikesaka. AP1000 architecture and performance of LU decomposition. In *Proc. Internat. Symp. on Supercomputing*, pages 46–55, 1991.

D.V. James, A.T. Laundrie, S. Gjessing, and G.S. Sohi. Scalable Coherent Interface. *IEEE Computer*, 23:74–77, 1990.

W.E. Nagel. Applications on the Cenju: First Experience with Effective Performance. In *2nd Cenju Workshop*, 1998.

Web page for the NAS Parallel benchmarks NPB2, 1997. `http://science.nas.nasa.gov/Software/NPB/`.

M. Snir, S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra. *MPI: The Complete Reference, Vol. 1, The MPI Core.* MIT Press, 1998.

A.J. van der Steen. Exploring VLIW: Benchmark tests on a Multiflow TRACE 14/300. Technical Report TR-31, Academic Computing Centre Utrecht, 1990.

A.J. van der Steen. The benchmark of the EuroBen Group. *Parallel Computing*, 17:1211–1221, 1991.

A.J. van der Steen, editor. *Aspects of computational science.* 1995.

A.J. van der Steen. Benchmarking the Silicon Graphics Origin2000 System. Technical Report WFI-98-2, Dept. of Computational Physics, Utrecht University, 1998. The report can be downloaded from: `www.phys.uu.nl/~steen/euroben/reports/`.