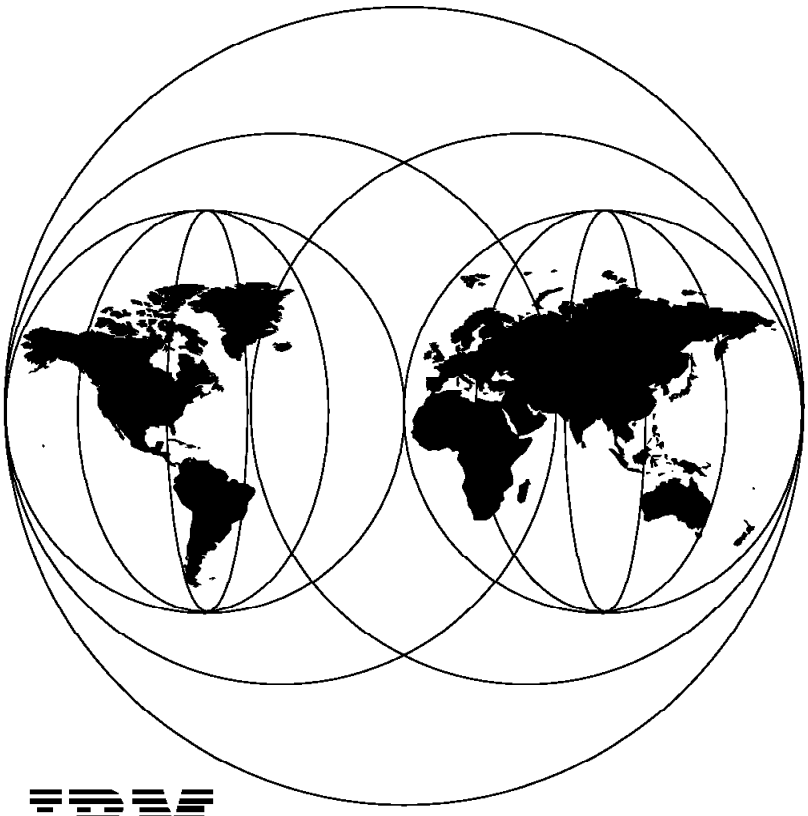
SG24-4810-00

Understanding IBM RS/6000 Performance and Sizing

February 1997



IBM

International Technical Support Organization

Austin Center

International Technical Support Organization

SG24-4810-00

Understanding IBM RS/6000 Performance and Sizing

February 1997

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix A, “Special Notices” on page 297.

First Edition (February 1997)

This edition applies to IBM RS/6000 for use with the AIX Operating System Version 4.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. JN9B Building 045 Internal Zip 2834
11400 Burnet Road
Austin, Texas 78758-3493

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

Preface

This redbook is intended to give guidance to system engineers, system administrators, and sales representatives on the following performance and sizing-related issues for IBM RS/6000 and AIX V4:

- Factors that influence system performance.
- RS/6000 architecture and hardware features relevant for performance
- Standard benchmarks
- How to size a system for optimal performance
- Performance tools and aids
- Sources of information on IBM RS/6000 performance

Some knowledge of IBM RS/6000 and the AIX operating system is assumed.

How This Redbook Is Organized

This redbook is organized as follows:

- Chapter 1, “Introduction” on page 1

  This provides a short introduction to performance considerations and explains the difference between meaningless and meaningful performance indicators .

- Chapter 2, “Background” on page 5

  This provides a review of the major theoretical notions that are useful when considering performance or sizing of RS/6000 machines. Also included is a description of some aspects of 64-bit technology and how IBM participates through AIX and RS/6000 technology.

- Chapter 3, “IBM RS/6000 Architectures” on page 29

  This provides an overview of the different RS/6000 architectures, like POWER, POWER2, and PowerPC.

- Chapter 4, “Hardware” on page 47

  This provides relevant information on processor and memory features and hardware products such as disk storage, asynchronous communication adapters, LAN/WAN adapters, and graphics adapters.

- Chapter 5, “IBM RS/6000 Products” on page 107

  This provides information on IBM SMP and IBM SP products and their performance-related features. An SMP-versus-SP selection guide also is provided in this chapter.

- Chapter 6, “Benchmarks” on page 149

  This provides an overview of several industry benchmarks such as SPEC benchmarks, transaction-oriented benchmarks (TPC), WebStone, NotesBench, and others. For each benchmark, a description, its metrics, and its usage is included.

- Chapter 7, “Sizing” on page 179

This provides general sizing concepts as well as particular hints and tips for configuring and sizing several different IBM RS/6000 environments.

- Chapter 8, “Performance Tools” on page 259

  This provides an introduction to performance monitoring concepts and tools. Each tool is briefly described, and its purpose and use is given. Besides the AIX performance tools, products like Performance Toolbox for AIX, Performance Reporter, BEST/1, and NT Performance Monitor are described.

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization Austin Center.

Andreas Hoetzel is an International Technical Support Specialist for RS/6000 and AIX Performance at the International Technical Support Organization, Austin Center. He writes extensively and teaches IBM classes worldwide on all areas of AIX internals, performance, and tuning. Before joining the ITSO, Andreas Hoetzel worked in the AIX Competence Center in Munich, Germany, as an AIX Technical Support Specialist.

Karine Cousergue is an AIX System Engineer in France. She has three years of experience in the field. She has worked at IBM for three years. Her areas of expertise include HACMP and Lotus Notes.

Diana Gfroerer is an AIX Technical Support Specialist in Munich, Germany. She has four years of experience in AIX. Her areas of expertise focus on monitoring and analyzing AIX system performance and AIX system tuning.

Frank Queau is a Service Specialist in France. He has six years of experience with large-industry customers. He has worked at IBM for six years. His areas of expertise include AIX performance and system management.

Warren Serrano is an AIX support specialist in Costa Rica. He has nine years of experience with UNIX operating systems. He has worked at IBM for one year.

Djony Tanuwidjaja is an RS/6000 Marketing and Sales Support Specialist in Indonesia. He has six years of experience in the RS/6000 and AIX field. He has worked at IBM for six years. His areas of expertise include benchmarking and performance tuning.

Gerardo Valencia is an IT Specialist in Colombia. He has four years of experience in RS/6000 and AIX. His areas of expertise include AIX performance and system management.

Thanks to the following people for their invaluable contributions to this project:

Yves Bex
International Technical Support Organization, Austin Center

Al Mitchell
International Technical Support Organization, Austin Center

Joanne Luedtke
International Technical Support Organization, Austin Center

Marcus Brewer
International Technical Support Organization, Austin Center

Stephen Riggs
International Technical Support Organization, Austin Center

Rebeca Rodriguez
International Technical Support Organization, Austin Center

Heidemarie Hoetzel
IBM Germany

Zoran Gagic
IBM Australia

Jon Gittoes
IBM UK

Hartmut Sprandl
IBM Germany

J. B. Moulin
IBM France

Philippe Ebert
IBM France

Bret Olszewski
IBM Austin

Kathy Cacciatore
IBM Austin

Gary Wiseman
IBM Austin

Comments Welcome

We want our redbooks to be as helpful as possible. Should you have any comments about this or other redbooks, please send us a note at the following address:

redbook@vnet.ibm.com

Your comments are important to us!

Chapter 1. Introduction

Performance must be considered within a given context. A good performance in the theater is different from good performance in a computer system. The latter usually means that the system responds to user requests in an acceptable time. This can mean anything from microseconds in real-time systems to hours for very large numeric-intensive computing jobs.

In the multiuser commercial environment, good performance invariably means a transaction response time of a few seconds. A typical expectation in an Information Technology (IT) department might be: ″The system shall deliver a response time of 2 seconds or less at the user's terminal for 90 percent or more of the transactions at peak times″.

In this example, the IT department then has to decide on the configuration which will be needed to fulfill this expectation. How can this be done? There are a number of ways:

1. Find a site using the same application software and make an estimate from the performance observed there.
2. Do a detailed simulation with either real users or a remote terminal emulator (RTE). This might be done with the actual application software or simple programs to generate the anticipated demand on the system′s resources.
3. If the system is not operational, a detailed walkthrough of the design specification can give an estimate of what resources the target system would need to handle the planned transaction workload.

In situations 2 and 3, you would need to gather data on what resources are consumed. This may seem complicated, since there may be different transaction types and other workloads. However, just as all matter is made up of protons, neutrons and electrons, all workloads are made up of:

- CPU resources consumed
- Memory resources consumed
- I/O load
- Network load

By decomposing a given workload into these basic elements, it′s possible to estimate the CPU, main memory, disk, and network resources needed to fulfill the response time requirements. The easiest way to assure good response times is to configure enough resources to ensure there will never be a shortage, but this may not be practical. Another method is to maximize use of the resources on a given system. However, simple queuing theory shows that this can result in extremely poor response times, since some elements of the system rapidly deteriorate if their usage exceeds a certain limit. It is these limits that need to be investigated if we are to design a system that will remain within them.

One needs to understand the architectures, implementations, performance, and limits of these elements to be able to design a system that performs to the set requirements.

1.1 Meaningless Indicators of Performance

In the original announcement material for the RS/6000 family, IBM, like most other companies in the workstation marketplace, quantified the raw performance of the RS/6000 processor line using millions of instructions per second (MIPS). MIPS is one of the most popular yet misunderstood measurements for comparing CPU performance. Many people jokingly define MIPS as ″Meaningless Indicator of Processor Speed″. We will show you why they are doing this.

Contrary to popular belief, MIPS measurements have nothing to do with the number of instructions which can be executed by a given processor in one second. MIPS ratings are a relative measurement. They are measured by comparing the performance of the system being tested against a reference machine running the Dhrystone benchmark. This reference machine is a Digital Equipment VAX 11/780. This machine can perform 1757 Dhrystones per second. Thus a 2 MIPS machine is one that is able to perform twice as many Dhrystones per second as the reference machine.

The Dhrystone benchmark is a very small synthetic program whose code and text segments easily fit into their respective caches. The benchmark does not test memory or I/O bandwidth and issues no system calls. The benchmark is heavily lopsided toward string handling and spends more than one third of all CPU time in the two C functions `strcmp()` and `strcpy()`. Few real-life applications behave this way.

To make matters even more complicated, there are two different versions of Dhrystone. IBM′s initial MIPS ratings were based on the Dhrystone 1.1 benchmark. Other workstation vendors eventually used the Dhrystone 2.1 benchmark as the basis for calculating their MIPS ratings. Dhrystone 2.1 had been modified to use two source files in order to defeat modern compiler optimization techniques such as ″loop unrolling″.

MIPS figures are derived from the Dhrystone run-time simply by dividing the Dhrystones per second by 1757 based on the assumption that the VAX 11/780 -- which performs at 1757 Dhrystones per second when using a certain C compiler -- should be a 1 MIPS machine. MIPS ratings for Dhrystone version 2.1 are calculated by dividing Dhrystones-per-second by 1650.

As workstations have evolved, other factors such as cache size and different processor configurations have made the single order of merit benchmarks such as Dhrystone obsolete. This has brought about industry standard tests such as the Systems Performance Evaluation Corporation (SPEC) suite of benchmarks.

So why are measurements with the Dhrystone benchmark and MIPS no longer really valid for comparing workstation performance?

There are numerous problems with the approach of quoting MIPS. Some of these are

- MIPS suggest a metric of hardware performance. The Dhrystone benchmark is highly susceptible to compiler optimization, yielding improvement of 100 percent or more. Making the assumption that compiler features benefiting the Dhrystone benchmark will accelerate other applications to the same extent is unjustified.

- Even if MIPS were an indication of hardware performance, the speed with which the CPU runs a given workload is highly dependent on the characteristics of that workload. For instance, when running a database application, the processor typically executes only about half the number of instructions per second as it might with technical or scientific applications.
- The Dhrystone benchmark does not represent the vast majority of real-life applications.
- Other factors such as cache or I/O performance may have a greater impact. A 10 MIPS processor which must wait nine CPU cycles while the data or instructions are accessed will process an application at the same speed as a 1 MIPS processor that has a steady stream of information and does not have to wait.

Processor speed, as indicated by a MIPS figure, is very misleading. The actual speed at which instructions are processed by the CPU may vary over a wide range. Care must be taken even with similar architectures.

It is clear that the MIPS figures taken at face value are not a fair or accurate method for comparing the RS/6000 with respect to the competition or even other RS/6000 models. For this reason, it has been decided by IBM that quoting MIPS figures may actually mislead customers and therefore should not be used as the basis for competitive bids.

1.2 Meaningful Indicators of Performance

As we have explained in the previous paragraph that the value of some traditional measures of performance are questionable, we will in the following try to explain the best methods and metrics for estimating and comparing workloads. Table 1 is a summary of some of the useful benchmarks reviewed in this book.

Application Type	Benchmark	Use
Single-user technical environments such as engineering or software development	SPECint95 SPECfp95	Tests CPU-intensive applications with heavy emphasis on floating-point/integer calculations.
Large, Multiuser/Commercial Environments	RTE/Custom Benchmarks	Tests workloads that are similar or identical to those anticipated by the customer. Can also be used for capacity planning and system tuning.
General Commercial	AIM	Tests real office automation applications. Tests memory management, integer and floating-point calculations, disk I/O, multitasking.
OLTP	TPC-C	TPC-C simulates network environments with a large number of attached terminals running complex workloads.
Decision Support Benchmark	TPC-D	TPC-D executes sets of queries against a standard database with large volumes of data and a high degree of complexity for answering critical business questions.
Graphics and CAD	PLBwire93 PLBsurf93 Viewperf Xmark93 X11perf	Using real applications (2-D design, 3-D wireframe, 3-D solid modeling, 3-D animation and low-end simulations), demonstrates relative performance across platforms/ systems.
NFS file systems	SPEC SFS	Measures the throughput supported by an NFS server for a given response time.
Web Server	SPECweb96 WebStone	Focuses on server performance and measures the ability of the server to service HTTP requests.
Lotus Notes	NotesBench	These six benchmarks measure the maximum number of users supported, the average response time and the number of Notes transactions per minute (called NotesMark).

Table 1. Meaningful Benchmarks

Chapter 2. Background

The objective of this chapter is to review the major theoretical notions that are useful when considering performance or sizing of RS/6000 machines. For the actual architecture and implementation of the RS/6000, refer to Chapter 3, "IBM RS/6000 Architectures" on page 29.

This chapter should not be viewed as a complete description of hardware architecture and UNIX systems. Only performance-related concepts will be presented.

2.1 Performance of Processors

First, let us take a look at the overall performance of a processor. It can be calculated like this (numbers are mean values):

$$\text{Execution Time} = \text{Number of Instructions} * \text{Number of Cycles per Instruction} * \text{Clock Cycle}$$

Figure 1. CPU Execution Time

The different factors affecting execution time are:

- **Number of instructions**

  The number of elementary operations needed to complete a program is a result of the compilation. This is called the path length.

- **Cycles per instruction**

  This number depends on the complexity of the instructions. The more complicated the instructions are, the higher the number of cycles consumed. But, on the other hand, there are fewer total instructions. This deals with the RISC/CISC concepts discussed in 2.2.1, "RISC/CISC Concepts" on page 6.

- **Clock cycle**

  The smaller the clock cycle, the faster the processor, but the more expensive its production cost! Indeed, to build a processor with high clock speed, costly improvements in the silicon technology need to be achieved.

All these different elements will be discussed in this chapter along with other performance-related topics.

2.2 Hardware Architecture

This part looks into the concepts of hardware architecture that are closely related to system performance.

2.2.1 RISC/CISC Concepts

Two different CPU designs have been implemented since the mid-'70s: CISC and RISC.

The first one, complex instruction-set computer (CISC), is the traditional design featuring a large and highly functional instruction set (more than 200 instructions). The instructions need several cycles to complete.

The need for complex instructions existed because, at that time, computers were equipped with small quantities of slow RAM. So, as complex instructions result in fewer instructions per program, one has to get less data from memory. But studies showed that only a small percentage of CISC instructions (around 10 percent) were commonly utilized by programs.

Later, as progress in semiconductor technology began to reduce the difference in speed between memory and processor, and as high-level languages replaced assembly language, the major advantages of CISC decreased.

The reduced instruction-set computer (RISC) concept was first defined by IBM Fellow John Cocke in 1974. It has some basic characteristics:

- A very simple architecture with an optimized set of machine instructions

  The instruction set consists only of elementary operations (less than 100 instructions) to reduce the complexity of the instruction decoder. Therefore, the CPU can execute with maximum speed and efficiency. The software generates other, more complex operations by combining several simple machine instructions. All these instructions have a fixed length (necessary for superscalar architecture, as seen later in 2.2.2, "Superscalar Architecture: Pipeline & Parallelism" on page 7).

- A very high instruction execution rate

  The objective of the RISC architecture is to be able to execute an average of one instruction per machine cycle. The execution time can be reduced to less than one using the superscalar architecture, as explained in 2.2.2, "Superscalar Architecture: Pipeline & Parallelism" on page 7.

- Compiler optimization

  The performance of the RISC architecture heavily depends on the compiler optimization. The compiler has to be able to exploit the hardware architecture by generating instruction sequences that take advantage of the capabilities and performance of the processor.

- Load/store architecture

  Memory access is separated from data manipulations in RISC architectures, so that the CPU is not stalled by slow memory access. Data is prefetched into registers, and then instructions only work with registers, which are the fastest memory available. Working with registers also allows the compiler to better organize data fetching according to data dependency.

In comparison, CISC tries to reduce the number of instructions for a program, whereas RISC tries to reduce the cycles per instruction.

Nowadays, both of these designs have evolved, and RISC architectures -- which are commonly utilized in the UNIX world -- have more than a hundred instructions. More important, RISC architectures are benefitting from the superscalar concept.

2.2.2 Superscalar Architecture: Pipeline & Parallelism

A pipeline is a hardware feature, similar to an assembly line, designed to increase instruction throughput through internal parallelism. Different units of the CPU perform, in parallel, the various operations required for fetching, decoding, and executing instructions. Several instructions can be executed in the CPU at the same time. The instructions go along the pipeline stages, in synchronization with the CPU clock. This means that, if everything goes well, each time a new instruction enters the pipeline, an older one is exiting it. This results in one instruction per pipeline and per cycle. Thus, although the time it takes to complete each instruction is not directly affected, pipelining increases the overall rate at which instructions complete.

When pipelining works as intended, performance is optimized. However, there are some potential problems: branch instructions and data conflicts. A pipeline normally holds a number of instructions in different stages of execution. Consider the case where one of these is a conditional branch, dependent on the condition code to be produced by a not-yet-executed instruction coming through the pipeline. Should it later turn out that the branch is to be taken, the system has to discard all the instructions prefetched after the branch and continue from the branch target address instead. A ″bubble″ in the pipeline will develop, leading to wasted CPU cycles.

A true data dependency arises when an instruction entering the pipeline needs the result still to be produced by an instruction further ahead in the pipeline. This case cannot be resolved by register renaming, the technique employed to avoid data conflicts. The younger instruction simply has to wait on the older one to produce the result.

While true data conflicts are uncommon, branches are frequently encountered. In fact, branch instructions constitute about 20 percent of the instructions in most computer architectures. Branch target prediction as used in the RS/6000 alleviates the problem to a certain degree. The basic problem that remains is that very complex software, like kernel code and database systems, suffers a slowdown of CPU speed in the pipeline because of the high percentage of conditional branch instructions that are typical for these environments. Simpler applications are less affected by this problem.

Next, came the idea of making several pipelines in order to implement further parallelism. This is called superscalar architecture. The instructions had to be distributed between the different pipelines and no more sequential treatment was possible. That is why compilers are so important in the RISC superscalar architecture: Complexity no longer lies in the instruction itself but in the compiler. But the advantage of a compiler is its ability to be optimized continuously, quickly, and much more easily than hardware code. Superscalar implies several independent execution units, like branch units, fixed-point units or floating-point units.

Superscalar allows more than one instruction to complete in a clock cycle. The objective is to achieve the highest number of instructions per cycle.

While the superscalar architecture aims at issuing more than one instruction per cycle, this goal is achieved only when the proper mix of instructions and data is sent through the pipeline. Some benchmarks will perform at several instructions per cycle, but the throughput might go down to less than one in other applications. This has nothing to do with instruction length, since the processor

can handle the large percentage of floating-point instructions typical of technical and scientific applications. Actually, this instruction mix promotes parallelism, since load and store operations and loop counting are handled by the fixed-point unit. The challenge for superscalar, highly pipelined RISC architectures lies in complex commercial applications that use the fixed-point and branch units only. These applications tend to have very short sequential execution paths and poor locality (as discussed in 2.2.3.2, “Locality Concept” on page 10).

Simplified Scheme of Superscalar CPU Architecture: Figure 2 shows a model of a three-pipelined architecture, the independent processor units being the branch, the fixed-point, and the floating-point processor units.

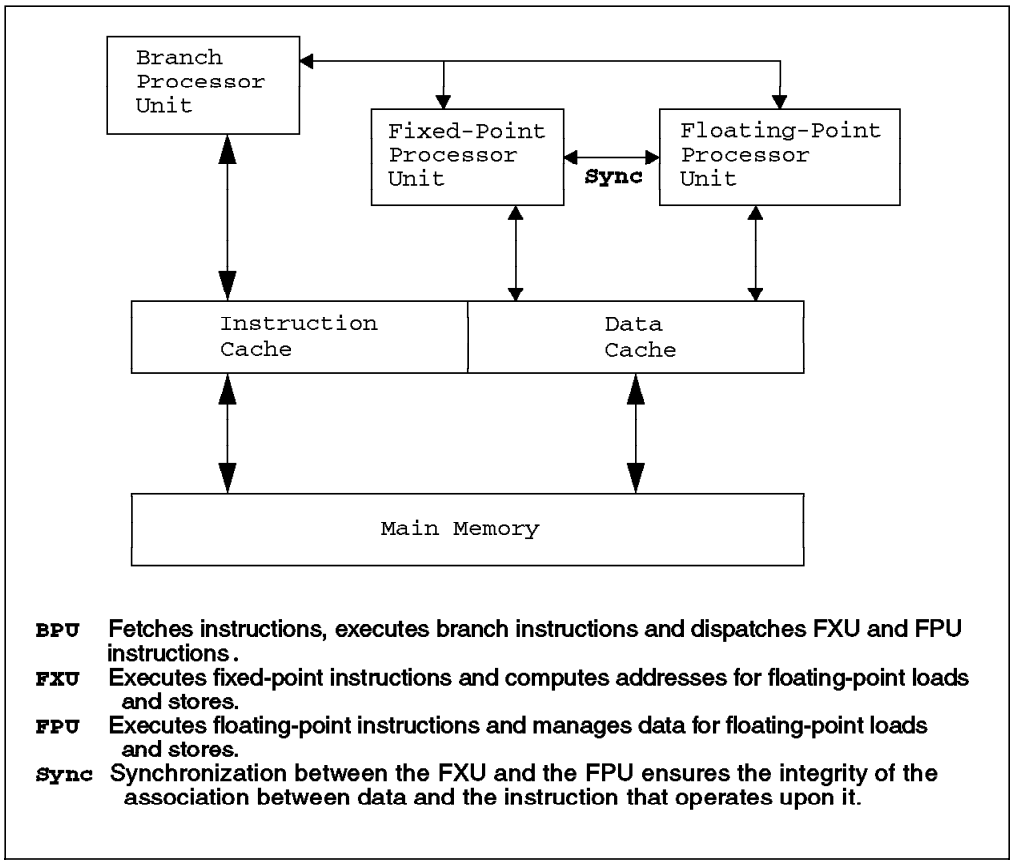


Figure 2. Pipelined Architecture

2.2.3 Memory Management

Efficient memory management can increase system performance. There are several layers and concepts involved.

2.2.3.1 Memory Hierarchy

Memory hierarchy is often referred to as four levels, spreading from disk to CPU.

But in advanced microprocessor architectures, this scale can be extended up to six levels, including cache levels L2 and L3.

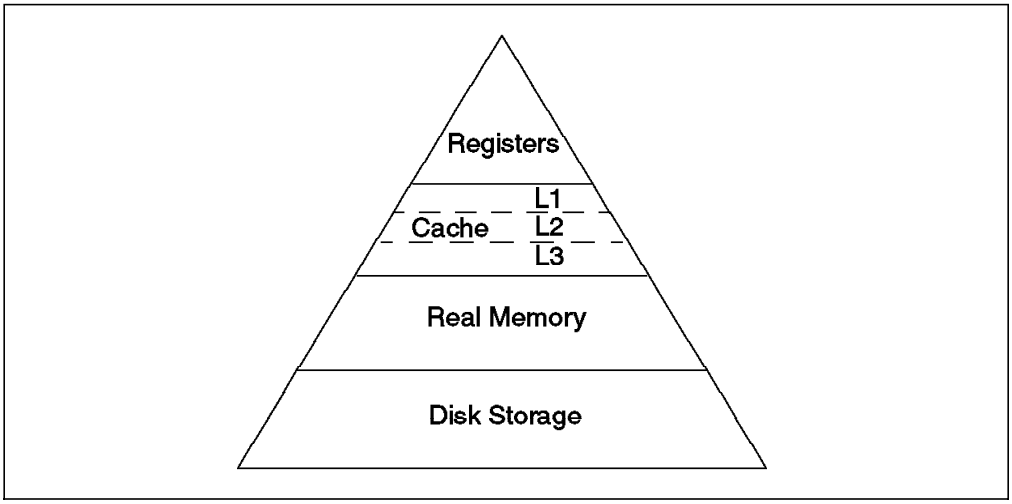


Figure 3. Memory Hierarchy

Each level in the pyramid is scarcer and more expensive than the one below.

- **Registers**

  Registers are storage cells within the specialized units inside the CPU pipelines. This is the fastest memory available, but there are only a few registers. (For example, the POWER2 architecture has 32 general-purpose registers and 32 floating-point registers. For more information, see Chapter 3, “IBM RS/6000 Architectures” on page 29.) Access is immediate.

- **Cache**

  Cache is a high-speed memory containing only a subset of main memory. This element is of great importance regarding performance considerations. Indeed, if the CPU accesses it instead of main memory for the most-frequently utilized instructions and data, it will gain many clock cycles.

  There are three different types of cache, levels 1, 2 and 3.

  L1 cache is located next to the pipelines and is the smallest. It usually is on-chip.

  L2 cache is usually outside the processor. It is a superset of L1 cache.

  L3 cache storage capacity is bigger than that of L2, but its access time is slower. It is a superset of L2 cache. When L3 is implemented, L1 cache generally is put on the chip for performance reasons.

- **Real memory**

  If the data is not in the cache, the data is fetched from main memory.

- **Disk**

  If the data is not in main memory, a page fault is taken (see Chapter 4, “Hardware” on page 47) and the data is retrieved from hard disk. This is by far the slowest way to get data.

Table 2 on page 10 summarizes typical measures made on five levels of memory. These numbers are only guidelines; they do not reflect the actual access times.

	Registers	Cache L1	Cache L2	Main Memory	Disk
Access time (clock cycles)	1	1	7-10	20-50	750.000-1,5 M
RS/6000 Storage Capacity	32 - 64 KB	16 - 288 KB	0.5 - 2 MB	32 MB - 2 GB	1 GB -thousands of GB

Table 2. Memory Access Times

2.2.3.2 Locality Concept

One of the basic principles defining how hardware and software interact is the concept of locality. Hardware expects that programs will exhibit patterns of address reference that are local both in time and space. To put it another way, it is assumed that programs access instructions and data according to the following models:

Locality in Time This means that, if an address is referenced, it is likely that it will be referenced again soon.

Locality in Space This implies that, if an address is referenced, it is likely that nearby addresses will also be accessed in the near future.

The principle of locality has given rise to the concept of working sets. The working set of a process is the collection of memory addresses that the process is currently using. This means addresses that the process has recently referenced or is likely to use in the near future. The working set thus comprises those memory ranges that the process needs to have access to without any significant delay in order to achieve maximum performance.

Inherent in the concept of working set is the observation that active address ranges normally do not shift gradually but rather tend to be replaced entirely in phase transitions. Most programs behave so that they stay around in one area of memory for some time, then suddenly move to another area, then stay around there for some time, and so on.

Although locality and working sets are rather vague concepts, based on empirical observations rather than strict laws, they are the rationale behind two very powerful architectural features of today′s computers: caches and virtual memory.

2.2.3.3 Cache

As explained before, cache memory sits between the CPU and main memory. The cache memory is nowadays typically divided into two sections, one for data (D-cache) and one for instructions (I-cache). In this way, for example, while the arithmetic units work on numeric data in the data cache, the branch processor can simultaneously load new instructions from the instruction cache, which increases parallelism.

Caches exploit locality on a smaller scale and offer much faster access times than main memory or disk.

Cache can be either integrated within the memory management unit (MMU) or located outside the processor (this is called external cache). Most modern RISC architectures now implement both internal and external caches, in order to

reduce access to main memory by having a bigger global cache size. In terms of performance, the nearer to the pipelines the cache gets, the smaller the access time is.

Data Organization: As cache only contains a subset of main memory data, its data need to be referenced for the CPU to find it.

Data is organized in lines because, otherwise, too much space would be used to reference each byte. So, each line begins with a tag containing the main memory address of the first byte and some control information like the valid bit. Then comes the real data, made of contiguous words.

When a cache miss happens, the whole line needs to be fetched from memory, as there is only one tag to reference the line.

The line size has some consequences on performance. Indeed, if you choose a small line size like 32 bytes, then a higher percentage of cache space is occupied by tags. This results in a smaller amount of cache, but data transfers between cache and main memory are almost immediate. On the other hand, if you choose a long line size like 128 or 256 bytes, it results in a larger amount of cache available for data, but transfers between main memory and cache are slower, because you need to fetch the whole line from main memory. For this kind of implementation, dividing a line into several sublines, each independent and with its own valid bit, may improve transfer time.

Hit Ratio: Among the various factors influencing performance, one of the most important in determining processor throughput is the **cache hit-to-miss ratio**. To achieve optimal performance, the CPU needs to achieve a high percentage of cache hits, meaning that the instructions or data required are present in cache memory. If not, then the processor will have to wait for the information to be loaded from main memory, which implies a performance degradation of as much as 50 percent. Effectively, while page faults cause either context switches or I/O waits, cache misses actually force the CPU into a wait state, and this forces idling while the requested data or instructions are fetched from memory or, in the worst case, disk.

The CPU wait state is forced because access to real memory is slower than access to the caches by more than an order of magnitude. Furthermore, the RISC architecture and the highly sophisticated pipelines found in the RS/6000 design work at top efficiency only when they can access code and data at a rate of two to four words per CPU cycle.

In addition to the cache hit-to-miss ratio, there is another important factor to be considered: the miss penalty, defined as the number of cycles the CPU must wait while the cache miss is being resolved by the memory subsystem. The cost of a cache miss, in terms of performance, is the product of the cache miss ratio and the miss penalty.

In general, the instruction cache is smaller than the data cache because programs are typically executed in chunks of four to five sequential instructions before the next branch instruction is encountered. Also, the hit rate is usually higher and the average access is faster than for the data cache because the instruction cache is never written to, and the consequences of a cache miss are more severe because the CPU pipelines are immediately stalled.

Cache Access: The first goal of a cache is to access data faster than memory. Therefore, cache searching must be very quick and efficient.

Generally, it utilizes a hashing algorithm to index the CPU addresses to locations in the cache (except for fully associative caches). Hashing implies that different CPU addresses can have the same index. The cache line tags with this index will then have to be compared to the CPU address to find out if it's a hit or a miss. The hashing algorithm has been chosen because it's an efficient way of limiting the search to only a few lines (the ones that refer to the same index).

Several cache organizations can be thought of:

- **direct mapped cache**

  The index refers to only one line of the cache where data may be stored. This is the simplest organization. However, as hashing will produce the same index for many different addresses, it can end up in cache thrashing. This happens when the same lines are continuously replaced by new ones before being reused.

- **n-way set associative cache**

  This organization is aimed at reducing the probability of cache thrashing. The idea is to group several lines (n) and to refer to them with one index. Each line is independent of the others in its set and has its own tag. Thus, when the CPU looks for an index, it has just n tags to compare to its own address. These comparisons are made in parallel, in order to not reduce performance. Cache thrashing is less likely, as several lines are provided for each index.

- **fully associative cache**

  This is a particular case of the preceding organization, when n equals the total number of lines in the cache. It means that there is only one set of lines. So no hashing is implemented. All the lines are looked through in parallel for each search. This is the most expensive cache organization. That explains why it is used only for small caches such as translation lookaside buffers (TLB).

When new data has to come into the cache, some existing line or subline must be put aside. This replacement policy, by which data is selected for removal, is usually done according to the least recently used (LRU) algorithm, which is easier to implement than techniques used for main memory (page aging, etc.).

Another extremely important policy is the update policy. The CPU has to store data. It can do it either to main memory or to cache. If the latter option is chosen, it increases the cache hit ratio because of locality, and the store time is decreased. That is why, in most cases, CPUs store data to cache.

But to ensure data integrity, cache needs to be consistent with main memory. Two options exist. First, write the data both to cache and memory. This is called the write-through policy. The advantage is complete coherency with memory. But it ignores the locality concept and always wastes a memory cycle. The other policy, called the write-back policy, asks the CPU to write only to cache. Data will be written to main memory just before it would be discarded (due to the replacement policy) or if the operating system requests it. Performance enhancement is quite clear, as fewer writes to main memory will occur, but this is done at the expense of main memory consistency. This policy is widely used throughout the different implementations.

Performance Considerations

- The bigger the cache is, the less main memory will be accessed.
- The write-back policy yields better performance than the write-through policy.
- For small caches, it is generally better to have large sets of lines so that the replacement policy will not induce too much cache thrashing.
- Due to spatial locality, the line size should be as large as possible. But very large line sizes will add some overhead when loading lines from memory.

2.2.3.4 Virtual Memory Concepts

The aim of this part is to define a few concepts that will be discussed in more detail in Chapter 4, “Hardware” on page 47.

Virtual memory has two technical meanings:

- The system can behave as though it has access to more physical memory than actually exists on the system. For example, a 32-bit system is limited to 4 GB of real memory (2 GB for AIX). However, AIX uses a virtual memory manager model that can support as much as 4 PB (4 Petabytes = 4,000 Terabytes) of virtual memory. This is accomplished by implementing a 52-bit virtual address.
- Process text and images are given effective addresses by the compiler, as opposed to real addresses. Since text and data have effective addresses, they can be loaded at any real memory location. Virtual memory allows many programs to occupy memory at the same time.

Swapping: Originally, UNIX systems used a technique called swapping to provide virtual memory. In a swapping environment, entire process images are loaded into real memory. Therefore, when a process is not needed in real memory (such as when it is sleeping), its image is transferred out to a secondary storage device. This secondary storage device is usually a disk partition known as the swap space. This swap space provides a backing store that allows the system to appear to have more physical memory than it actually has. The drawback to swapping is its slow mechanism, as the entire image of the process must be moved from real memory to swap space and back.

Paging: A newer virtual memory management technique is paging. In a paging environment, only the most popular pages of a process occupy memory at any given time. A page is a small chunk of code or data, having a fixed size throughout the system. For example, AIX Version 4 uses a 4 KB page size.

Like swapping, paging utilizes a secondary storage device, called the paging space, for backing store. When available real memory space for pages becomes scarce, the system moves the least popular (usually least-recently accessed) pages out of memory to the paging space.

Thus, it makes paging completely independent of any process.

There is also a hybrid approach to managing virtual memory. Paging is the standard method, but when real memory becomes overcommitted, the system begins to swap processes. Usually, only sleeping processes will be swapped out. The swapped out processes must then be put back into real memory before they can be made ready to run. This approach is utilized by AIX Version 4.

Performance Considerations: When dealing with memory, a couple of issues must be watched:

- Thrashing: The system spends more time handling page ins and page outs than performing computational tasks. Thrashing occurs when there is so much demand on the real memory that it becomes over-committed. It is a direct result of not having enough real memory to handle the workload. Thrashing is often characterized by a sudden slowdown of system response time and a large amount of disk activity.
- Running out of paging space: If not enough paging space is defined, it causes the kernel to prevent new processes from starting. The *SIGDANGER* signal is sent to most processes in alert. If the condition persists, the kernel may be forced to terminate processes.

2.2.3.5 I/O Buses

So far, we have reviewed notions dealing with the internals of a processor, cache, and memory management. Another performance-related factor that we need to consider is the I/O bus.

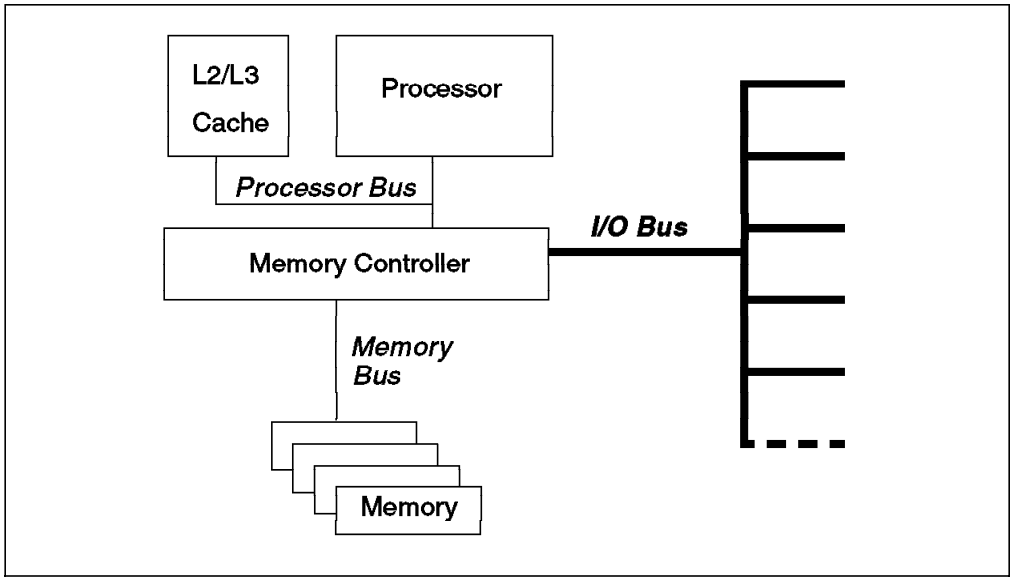


Figure 4. Simplified System Architecture: Focus on Buses

The major implementations of I/O buses are discussed below.

ISA Bus: Industry Standard Architecture (ISA) is the most widely used I/O bus in the PC industry. Originally, there were no official standards for it. Later on, its specifications were defined by the Institute of Electrical and Electronics Engineers (IEEE) standards group.

The ISA bus allows a transfer rate of up to 8.3 MB/s. Transfers over the ISA bus are synchronized around 8 MHz, and they usually take a minimum of two cycles of the bus clock to perform a data transfer. Since the data path of an ISA bus is 16 bits wide, up to two bytes may be transferred during each transaction.

The extended industry standard architecture (EISA) is a proper super-set of the ISA bus specification with better performance.

Support for EISA bus architecture was added to the RS/6000 family in 1995, first for the 40P workstation, and later for the 43P workstation and Exx, Fxx servers.

Micro Channel Architecture: IBM developed the Micro Channel architecture in the mid-'80s as a technology revolution from the ISA bus. Since the ISA bus has limited performance, lack of bus mastering capability, rows of jumpers and switches for setup, and poor electrical characteristics, this all made an alternative to the ISA bus necessary for high-performance desktop and server systems. The first release of the Micro Channel architecture supported basic 1-, 2- and 4-byte data transfers operating at 10 MHz. This yielded a peak 40 MB/s throughput for 4-byte transfers. In addition, the data transfers could be initiated by the processor or by bus master I/O devices that off-loaded work from the system processor. IBM later added streaming data capability to the Micro Channel architecture, providing even higher data rates. As the performance capability of the Micro Channel architecture evolved, this added capability was exploited by the new RS/6000 products. With peak performance capability of up to 160 MB/s (due to an increase in clock rate: 20 MHz), the Micro Channel architecture became more than just a viable alternative to ISA; it became an I/O expansion bus performance leader.

PCI Architecture: The Peripheral Component Interconnect (PCI) local bus specification was developed by the PCI Special Interest Group, led by a group of companies including Compaq, IBM, Intel, Digital and NCR. Introduced in 1992, the PCI bus architecture has quickly gained widespread industry acceptance.

The PCI bus is a clock-synchronous bus that runs at up to 33 MHz for standard operations, and it has an added capability to operate at up to 66 MHz. It can transfer either 32-bit or 64-bit data at a time. This yields a peak local bus performance of 132 MB/s for 32-bit transfer and 264 MB/s for 64-bit transfer at a clock speed of 33 MHz. The 66 MHz PCI capability could increase local bus performance to 528 MB/s for 64-bit transfers, but the PCI implementation in the RS/6000 is currently clocked at 33 MHz and uses 32-bit addressing.

PCI allows low-latency random access such that at 33 MHz, as little as 60 nanoseconds are required for a master on the bus to access a slave register. At 66 MHz, only 30 ns are required for this access.

The PCI bus was defined with a provision for processor architecture independence.

Summary of I/O Bus Capabilities

	ISA	EISA	Micro Channel	PCI
Data Path Width	16	32	16/32/64	32/64
Data Bus Speed (MHz)	8.33	8.33	10/20	33
Data Transfer Rate (MB/s)	16	33	20/40/80/160	132/264
Data Rate Implemented	16	33	80 (10 MHz) / 160 (20 MHz)	132
Number of Slots	0-8	0-8	0-8	0-4
Bus Masters Supported	No	Yes	Yes	Yes

Table 3. I/O Bus Capabilities

The problem when connecting the processor to the ISA/EISA bus directly is that the processor's speed has to be reduced to match the slow ISA/EISA bus speed.

Thus, the systems cannot take advantage of a fast processor. The solution to this problem is to use the PCI local bus as the primary system bus and the ISA/EISA bus as an expansion bus. This way, the system can take advantage of the high-speed data transfer provided by the PCI bus when communicating with the processor and memory. On the other side, through a PCI-ISA bridge, the bus clock can be reduced to match the ISA bus requirements.

2.2.4 MP Implementation Specifics

Different types of Multiprocessor (MP) technologies coexist. The three major ones are:

- **Shared Memory MP**

  A symmetric multiprocessor, also known as a shared memory or tightly coupled MP, has multiple processors that have their own cache and can each address the shared memory and all devices. User processes on any processor see the full machine. If two processors or more access the same word in memory, hardware keeps the caches consistent, invisible to application processes. Compared to other multiprocessor types, the advantage of SMPs is their use of the same programming model as uniprocessors.

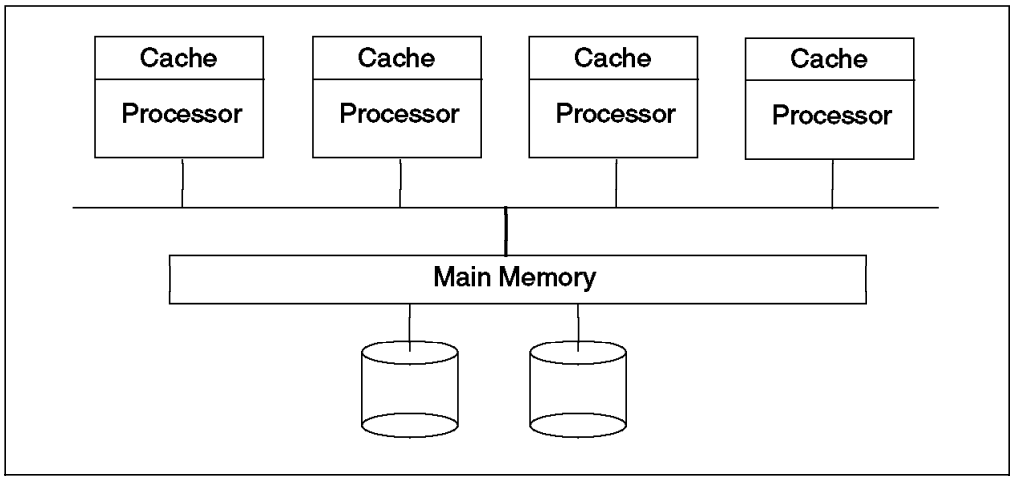


Figure 5. Shared Memory MP

- **Shared Nothing MP**

  All processors have their own memory and disks. Uniprocessor programs must be changed to use the parallelism of this configuration because they must pass messages across an interconnect in order to use the multiple processors.

  The IBM RS/6000 SP is an example of this kind of architecture.

  Shared nothing MPs generally scale better than SMPs because they have no memory bus contention and no cache coherency problems among the processors.

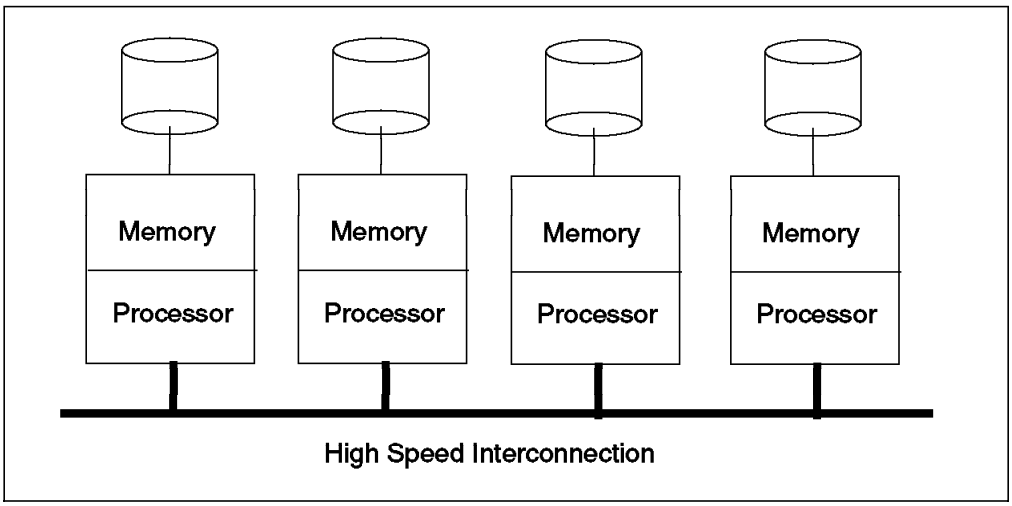


Figure 6. Shared Nothing MP

- **Shared Disk MP**

  Unlike the SMP, each processor on a shared disk multiprocessor has its own memory. That is why the shared disk multiprocessors, like the shared nothing multiprocessors, have no memory bus contention or cache coherency problems among the processors. However, a centralized locking scheme is used to control access to the disks. This locking scheme requires changes to some applications (such as databases) and generally offsets the performance advantages of no memory bus contention or the cache coherency problem.

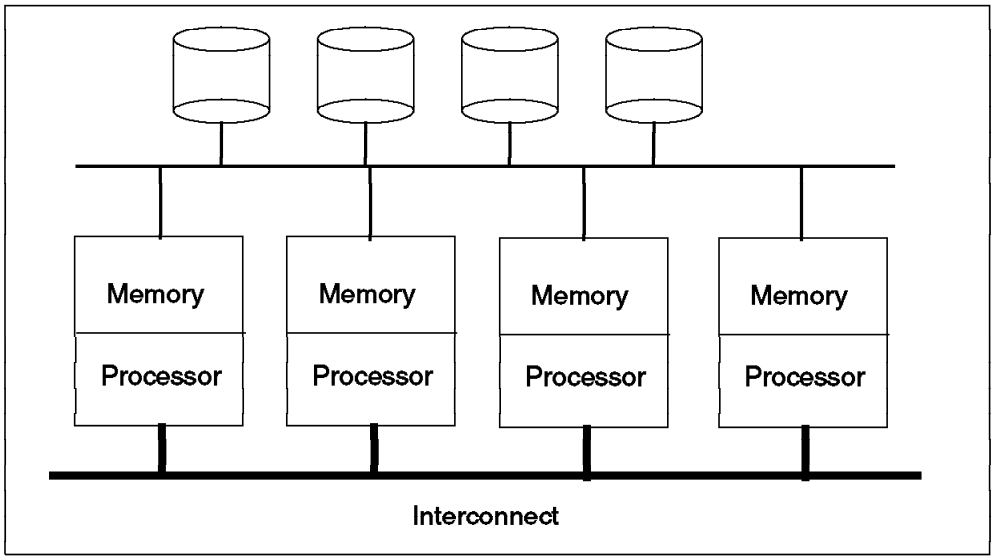


Figure 7. Shared Disk MP

2.3 The Kernel

UNIX is a multiuser and multitasking operating system. The kernel is the heart of the operating system.

2.3.1 Responsibilities

The kernel has two primary responsibilities:

1. Providing the I/O path between applications and the system hardware.

   I/O support includes device and file I/O. The kernel provides the mechanisms for the creation and management of files via file systems. The UNIX system establishes file names to represent logical and physical devices. These file system abstractions are found, by convention, in the /dev directory. This concept allows applications to access devices for I/O as if they were ordinary files. Thus, the file system provides the application interface for device I/O.

2. Handling the loading and execution of programs.

   In UNIX, an executable file is called a program. When a program is executed, its running image is referred to as a process.

   Since UNIX is a time-sharing operating system, the kernel implements scheduling routines to fairly allocate processor time slices to processes.

   The kernel is also responsible for allocating and de-allocating virtual memory for all active processes, as well as for providing mechanisms for interprocess communications.

2.3.2 AIX Characteristics

There are a few characteristics of AIX Version 4 that differentiate it from more-traditional UNIX kernels.

The AIX Version 4 Kernel is Preemptable: Traditional UNIX kernels allow processes that are running in kernel mode to finish their system calls before preempting them to allow other processes to run. While this method provides a simplified scheme for synchronizing access to data structures, time-critical applications may become blocked waiting for a less-favored process to complete its system call.

AIX Version 4 permits processes to be preempted even if they are in the midst of a system call. Kernel locks are provided to safeguard kernel data integrity.

The AIX Version 4 Kernel is Pageable: This is unlike many UNIX systems.

It means that portions of the kernel can be paged out to paging space. This allows more real memory for applications. Of course, some critical portions of the AIX kernel, such as interrupt handler code and data, are pinned in memory. A pageable kernel means that the system time needed for a system call may vary, depending on whether the pages called are in memory or in the paging space.

Kernel Extensions: AIX defines kernel extensions as entities added to the base kernel. These extensions can be added to the kernel dynamically, without the need to reboot the system.

Kernel extensions are:

- device drivers
- system calls
- virtual file systems (journaled file system, network file system, CD-ROM file system)
- streams modules (an AT&T creation that facilitates the creation and implementation of character I/O mechanisms)

Executable File Formats: AIX defines the format of a compiled, executable file as Extended Common Object File Format (XCOFF). XCOFF is based on the AT&T definition of COFF. Programs compiled by the AIX C, C++, Pascal or Fortran compilers generate XCOFF files. One of the major interests in XCOFF is its ability to dynamically resolve references to shared libraries and other external objects. COFF, on the other hand, can only resolve references statically.

Kernel Subsytems: Figure 8 illustrates the relationship between kernel subsytems.

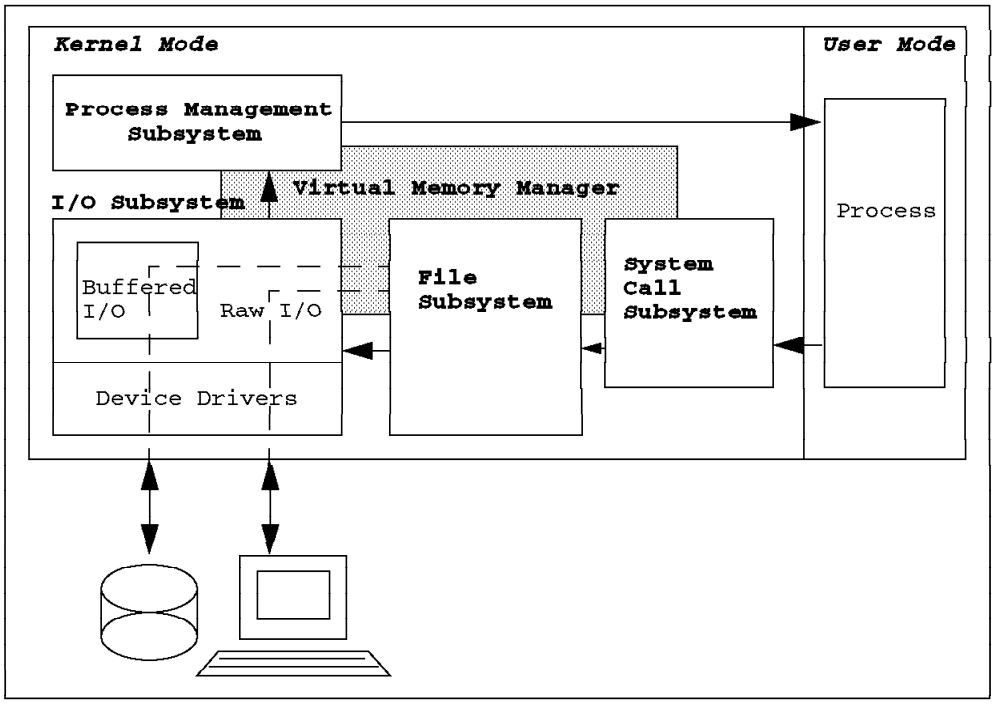


Figure 8. AIX Version 4 Kernel Subsystems

When the system is executing user code, it is said to be running in **user mode.** When the system is executing system calls or other kernel code, such as interrupt handlers in the device drivers, it is said to be running in kernel mode.

Processes access the I/O and process management subsystems via the system call subsystem. The process management subsystem is responsible for scheduling and dispatching processes. The kernel uses two types of interfaces to devices, buffered I/O and raw I/O. Buffered I/O (like hard disk drives or floppy diskette drives) is performed in blocks of data. The blocking factors and schemes used by block devices are controlled by device drivers. Raw devices (such as printers or terminals) perform I/O one character at a time. The virtual memory manager supports both process management and device and file I/O.

2.3.3 Context/Thread Switches

In AIX Version 4, the scheduled entity is the thread (as opposed to process in Version 3). There are 128 levels of priority (0-127). The lower the number, the higher the thread priority. Each second, the scheduler recalculates the priorities. Each tick (time slice), the dispatcher chooses the thread to run.

A context switch occurs when the execution of the current process is stopped and replaced by the next one (determined by the scheduling policy). The system must store the state and data of the current process and then load those of the next process to be executed.

A thread switch is the corresponding action when dealing with threads inside the same process. Because the same process is still executing, less information needs to be safely stored before switching to the next thread. Therefore, this mechanism is much faster than the context switch.

2.4 64-bit Architecture

Microprocessor technology changes very rapidly in terms of performance, chip size and capability. There are several major evolutions in this technology, such as CISC-to-RISC architecture, followed by 32-bit-to-64-bit technology. There is a perception that by using 64-bit technology, everything will be faster, and most people believe that 64-bit technology will solve existing 32-bit issues. This chapter will try to describe some aspects of 64-bit technology and how IBM participates through AIX and RS/6000 technology.

2.4.1 Reasons for the 64-bit Architecture

Although the vast majority of the current applications do not need the functions and capabilities of a 64-bit architecture, more applications of the near future will view 32-bit technology as a limiting factor. In fact, a few applications requiring large memory and large files already exist.

There are several reasons to use 64-bit technology:

- It can perform arithmetic on 64-bit-long integers and floating-point numbers. This is possible within the CPU without software looping. It may be useful in some specialized applications, especially scientific or engineering applications, which need double-precision (64-bit floating-point) instead of single-precision (32-bit floating-point).
- It allows data structures larger than 2 GB. This eliminates the process size limitation of 2 GB. It may be useful in some scientific and data mining applications.
- It can address physical memory beyond 2 GB. (Theoretically, 4 GB of real memory can be addressed with 32-bit, but the limit for AIX is 2 GB.) It is obviously a system-wide advantage to have so much memory because of the large database that can be stored there. This is usually the aspect that everyone is interested in.

Some applications, including databases, imaging and multimedia, will benefit from the 64-bit architecture because they can access very large files and a lot of data. The ability to manage larger programs and data will be very important in the future. The 64-bit architecture should play a key role in this evolution. However, before different hardware vendors begin developing their own ideas of

what 64-bit systems should look like and how they should behave, it is vital that the industry defines and agrees upon standards. Otherwise, the resulting software incompatibilities would soon impact both customers' and software providers' ability to choose the system platform that best suits their needs or to easily move applications between platforms.

2.4.2 Types of Addresses

There are three types of addresses in a computer system:

1. Effective Address

   It is generated and utilized by machine instructions, and it governs the amount of addressing available to a given UNIX process. The effective address is specified by the architecture.

2. Virtual Address

   It governs the amount of addressing available to a given application. It is not specified by the architecture.

3. Physical Address

   It is the address of memory cards plugged into memory slots. It governs the amount of physical memory that can be addressed. It is specified by the architecture.

2.4.3 Understanding 64-bit Architecture

In defining 64-bit architecture from an operational standpoint, the following definitions apply:

- 64-bit architecture can handle 64-bit-long data. In other words, a contiguous block of 64 bits (8 bytes) in memory is defined as one of the elementary units that the CPU can handle. This means that the instruction set includes instructions for moving a 64-bit-long data set, as well as arithmetic instructions for working with 64-bit-long integers and floating-point operations.
- 64-bit architecture generates 64-bit-long addresses, both as effective addresses and as physical addresses. Individual processor implementations may generate shorter physical addresses, but the architecture must support lengths of up to 64-bits.

2.4.4 Understanding 64-bit Machines

In defining a 64-bit machine from the CPU standpoint, the following definitions apply:

- A 64-bit machine implements a 64-bit architecture. It can execute 64-bit instructions, handle 64-bit data as an elementary unit and allow for addressing more than 4 GB of physical memory.
- A 64-bit machine has 64-bit-wide (or wider) internal processor buses.

2.4.5 Addressability

Addressability is the most important aspect, as we expect that future, complex applications (large databases, large numeric applications, and multimedia environments) will need to manage and operate on larger data sets. Table 4 on page 22 shows the size of the address space that can be managed as a function

of the length of the address that the CPU generates. A 64-bit architecture can address a huge address space.

Address Length	Address Space
8 bit	256 bytes
16 bit	64 kilobytes
32 bit	4 gigabytes
52 bit	4,000 terabytes
64 bit	16,384,000 terabytes

Table 4. Size of Address Space

2.4.6 CPU, System and I/O Bus

In a microprocessor, the internal buses are usually defined as the physical interfaces that interconnect the different operational units on the chip. In order to be properly defined as a 64-bit machine, a processor has to be equipped with 64-bit (or larger) internal buses. If these buses were narrower than the word size of the processor, they would become a performance bottleneck, making it impossible for the CPU to run at full speed. For this reason, the 64-bit implementations of the PowerPC provide internal bus widths of 64, 128, or more bits.

The same concepts apply to the system bus (the bus that typically connects the CPU[s] to the memory subsystem and the I/O controller) and the I/O bus (the bus that moves data between the CPU complex and the I/O cards). Some RS/6000 systems today are equipped with 128- or 256-bit-wide system buses, and their Micro Channel I/O buses move data in 64-bit-wide blocks. On the RS/6000 SMP models, the system bus is replaced by a crossbar switch that moves data at 1.8 GB per second. The introduction of 64-bit CPUs will further improve the characteristics and performance of these buses.

2.4.7 Advantages of 64-bit Architecture

Larger Programs: Figure 9 on page 23 shows that a 64-bit processor can support a much wider address space when compared to its 32-bit equivalent. This support means that it is possible to develop much larger and more complex applications, whereas today's 32-bit applications have a 4 GB size limit. In AIX, this limit is 2 GB. However, most current applications do not reach or even approach this limit. In fact, thousands of real-life applications still run on 16-bit architectures and operating systems. Future applications, however, will need to go beyond the current 4 GB limit.

Larger Data: Another advantage of 64-bit architectures is their ability to manage very large amounts of data, possibly in real memory. This ability greatly simplifies the task for applications that typically handle big data sets, such as multimedia, statistics, or database applications.

Although 32-bit environments do not limit most existing applications, the demand for 64-bit capability will increase in the future, as more powerful and demanding applications develop. The ability to handle elementary data larger than 32-bit is also desirable, but many advanced applications still work on data smaller than

64-bit. For instance, multimedia sound requires 16-bit-long data, and graphics requires 24- and sometimes 32-bit-long data.

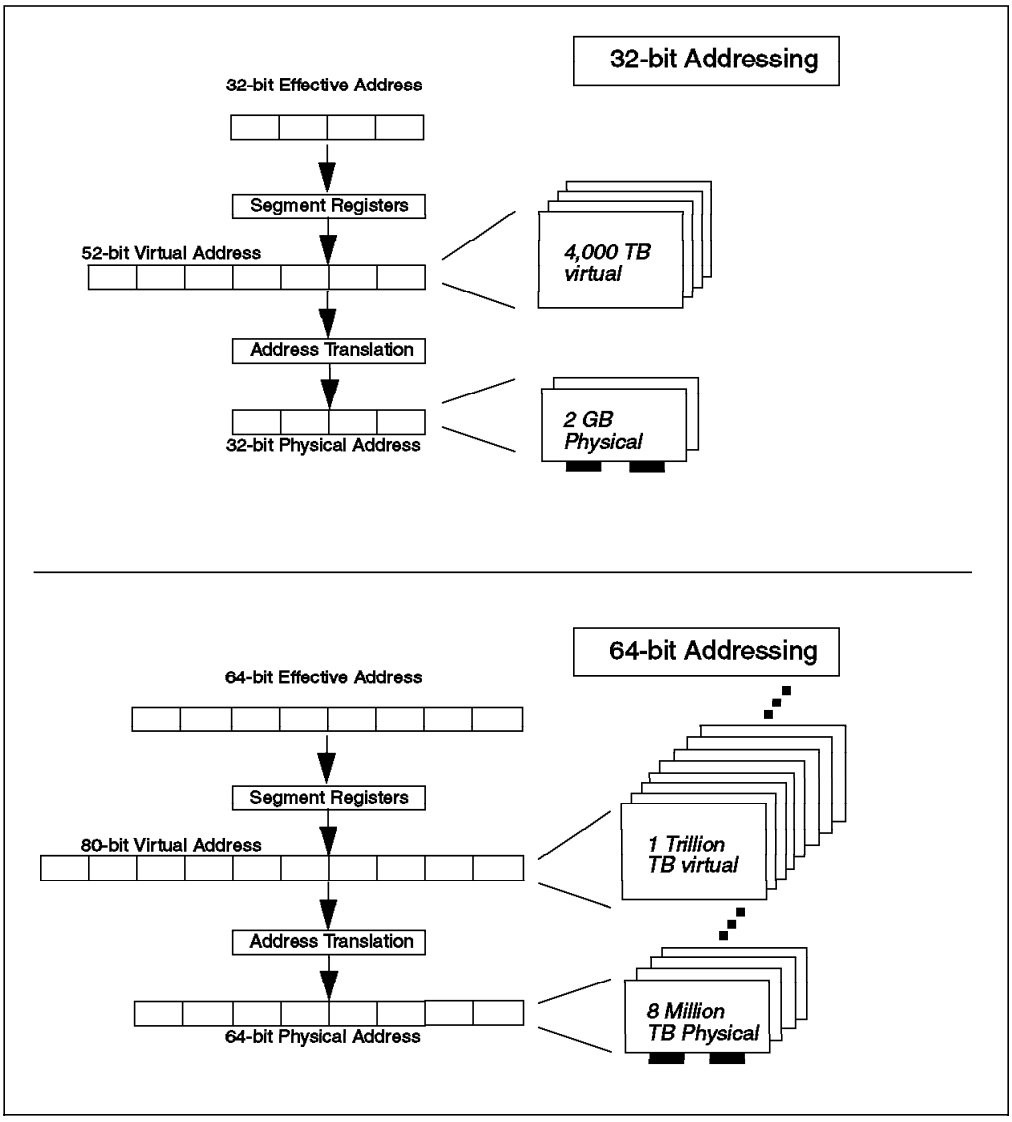


Figure 9. Virtual and Physical Memory Support of 32-bit and 64-bit PowerPCs

Larger Files: A 64-bit environment can easily manage huge files, file systems, and databases without forcing the application or the operating system to explicitly handle situations requiring data sets larger than 2 GB. Today's most sophisticated application enablers, such as DB2 for AIX, already support this capability.

AIX Version 4 itself allows for file systems larger than 2 GB. Advanced applications using high-quality graphics and sound or managing huge databases (such as weather forecasting, decision support, or virtual reality) are going to become more common, however, and they will greatly benefit from systems that can address and manage such huge amounts of data without forcing the application to worry about limits in data size.

Larger Physical Memory: The physical addresses that a 64-bit chip generates are up to 64 bits long as well. Once again, this size eliminates the 2 GB limit in real memory that all 32-bit architectures share. Actually, many of today's 32-bit

systems don't even support as much as 2 GB of real memory and have much lower physical limits. Not many current UNIX installations have that much memory, although hardware providers would love to have more. However, memory requirements of applications grow every year, and, in the future, real memories greater than 2 GB will become more common.

2.4.8 Performance of 64-bit Architecture

Although 64-bit architectures deliver all the benefits just described (larger programs, data, files, and physical memory), a common misunderstanding is that a 64-bit processor per se increases performance. This is not so for the following reasons:

- The larger address space means that larger applications can be developed and executed, without relation to performance.
- The support for larger physical memory gives no advantages if the memory cards are not there. Having large real memory for huge applications will reduce paging, an issue more related to software than to hardware.
- The ability to manage larger disk spaces is a great functional and programming advantage if the software is modified to exploit this ability. In itself, 64-bit hardware will not influence performance.
- A 64-bit architecture can do 64-bit arithmetic, so if an application needs this function, programmers can avoid writing a library. Again, this ability is an indirect, software related, performance improvement. It is of no benefit to most integer calculation, because text is 8-bit, audio is 16-bit, and full-color graphics is 24- or 32-bit. It may have advantages in video compression, multimedia and cryptology.
- It will double the size of every pointer and address, which makes the operating system and the programs larger. In other words, the memory requirement will be higher. This will impact cache, bus, and memory, creating demand for a higher memory bandwidth. This may actually decrease performance if it is not designed very well.

2.4.9 Software Considerations for 64-bit Architecture

Software or application exploitation in 64-bit systems is not automatic. Several things have to be considered:

- The operating system needs to support the 64-bit processor. This enables support for more than 4 GB memory.
- Compilers need to incorporate 64-bit support.
- Applications need to be 64-bit enabled if they want to take advantage of 64-bit architecture. Today, there are very few 64-bit applications. An existing 32-bit application has to be recompiled or ported to the 64-bit environment, but this will introduce big modifications. If the application is written in C language, there are several things to be taken care of, for example:
  - Pointer size in 64-bit environment is 64-bit.
  - C data structure will be aligned to 64-bit.
  - All ″int″ (integer) data will be 64-bit.

2.4.10 AIX and 64-bit Implementations of PowerPCs

In recent cooperative efforts, the most important UNIX hardware vendors ratified an initiative on 64-bit computing. On August 15, 1995, computer industry leaders supporting current architectures (Intel's Pentium, Hewlett-Packard's PA-RISC, Digital's Alpha, IBM's PowerPC, Silicon Graphics/MIPS RISC, and Sun's SPARC) endorsed the development of a common, 64-bit, UNIX API. Coupled with a common, 64-bit data representation model in the C programming language, the API set will support supercomputers, enterprise and workgroup servers, workstations, and networks.

Because widely accepted industry standards at the 64-bit level do not exist, the initiative gives industry leaders (UNIX suppliers, software application developers, and computer manufacturers) a unique opportunity to support an efficient 64-bit UNIX model. By building from existing 32-bit APIs, the 64-bit API will also be upward-compatible with 32-bit applications.

The 64-bit UNIX API specification, when accepted by the X/Open standards body, will comply with and track existing standards such as X/OPEN Portability Guide (XPG 4.2, a.k.a. Spec 1170), Portable Operating System Interface for UNIX (POSIX), System V Interface Definition (SVID), Common Desktop Environment (CDE), and X-Windows.

From the CPU standpoint, the PowerPC architecture is an open, extendable design. There is nothing in the chip architecture itself that would affect binary compatibility in migrating across 601, 603e, 604, 604e or 64-bit implementations. The PowerPC processor architecture was defined, from the start, as a 64-bit architecture that is a superset of the 32-bit architecture implemented in the 601, 603 and 604 processors. The PowerPC architecture defines 64-bit instructions and address translation, 64-bit-wide integer registers and 64-bit data transfer and arithmetic instructions. The PowerPC architecture has a 32-bit mode as a ″functional subset″, which is implemented in the 601, 603e, 604 and 604e.

One goal of the 64-bit version of the PowerPC is to maintain binary compatibility with the current PowerPC processors. From the standpoint of the 32-bit and 64-bit specifications, there are a few differences, as shown in Figure 10 on page 26. The number of CPU registers (the basic storage cell where the CPU stores the data on which it performs its computations) remains the same, but these registers are 64 bits long instead of 32 bits long. A few other control registers move from 32 to 64 bits in length. As shown, floating-point registers do not change in size, as they conform to industry standards for floating-points, which require 32 or 64-bit-long data.

In a 64-bit implementation of the PowerPC, existing machine instructions do not change much. Many instructions simply work in 64-bit mode. That is, they can manage 64-bit-long data and use/generate 64-bit-long addresses. New instructions, which were not implemented in the previous PowerPC chips, are included to handle 64-bit data.

A 64-bit PowerPC can also work in 32-bit mode. In this way, any 32-bit operating system or application that currently runs on the PowerPC can run unchanged. For example, arithmetic instructions running in 32-bit mode will operate on the lower half of the CPU register involved and, in the result, will consider only that half of the register. Such chips will handle 32-bit addresses in the same way.

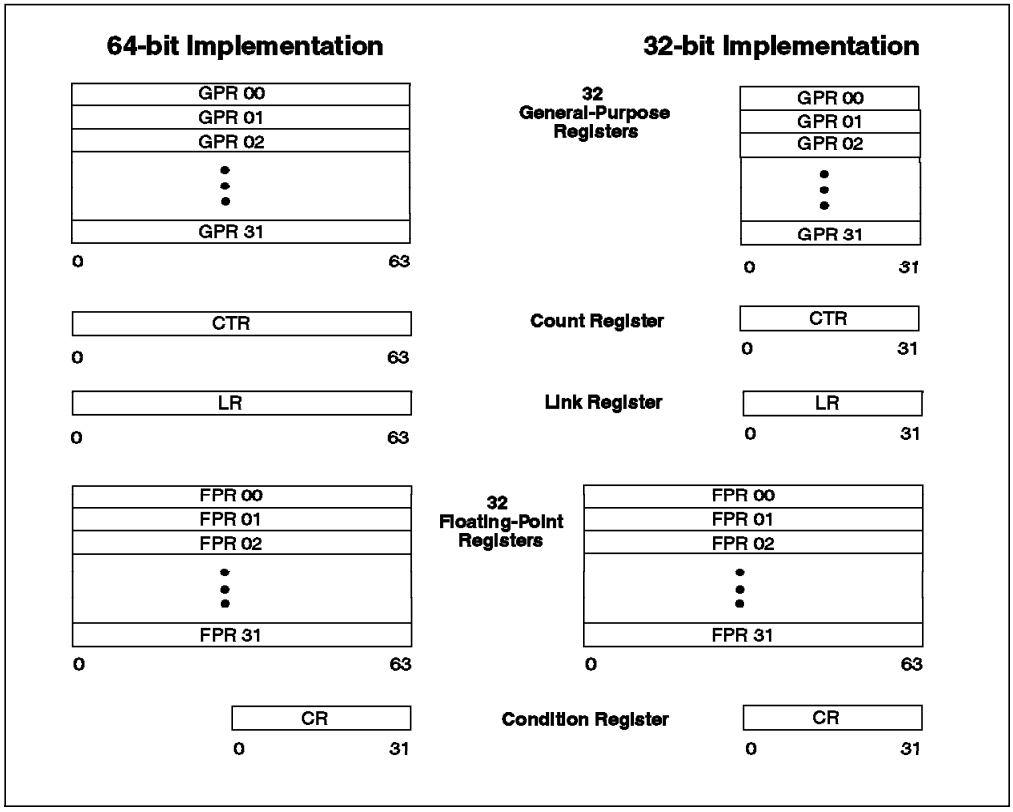


Figure 10. Comparison of 32-bit and 64-bit PowerPC Processors

AIX is currently being enhanced for future support of 64-bit RS/6000 systems. The AIX 64-bit support is designed to run 32-bit applications and 64-bit applications concurrently on 64-bit systems without interference with each other. AIX plans include full binary compatibility for 32-bit applications on 64-bit RS/6000 systems with no modification required.

2.4.11 Conclusion

The global performance of a machine does not depend only on the speed or characteristics of its microprocessor. It derives from a balanced, high-performance design in which all components contribute to the processing power of the system complex.

The latest technology looks very nice, but it may not give your business the benefit you expect if the integration between software (OS, application), hardware (CPU, memory, I/O) is not designed very well to achieve the best overall system performance.

2.5 References

http://fnctsrv0.chips.ibm.com/products/ppc

http://www.rs6000.ibm.com/resource/technology

http://www.rs6000.ibm.com/resource

UNIX Systems for Modern Architectures, Curt Schimmel, 1994

SMP Overview, Deborah Blakely-Fogel, AIXpert November 1994

PowerPC and POWER2 Technical Aspects of the New IBM RISC System/6000, IBM Corporation, SA23-2737

A Technical Introduction to PCI-Based RS/6000 Servers, SG24-4690

Managing AIX V4 on PCI-Based RISC System/6000 Workstations (40P/43P), SG24-2581

Chapter 3. IBM RS/6000 Architectures

In February 1990, IBM introduced the first RISC System/6000 (RS/6000) with the first Performance Optimization With Enhanced RISC (POWER) architecture. Since that date, several POWER architectures have been designed for the RS/6000 models.

In 1991, with the alliance of Apple and Motorola, IBM started a plan for the future that would span a range from the small, battery-operated computer to very large supercomputers and mainframes. The PowerPC family of microprocessors, a single-chip implementation jointly developed by Apple, IBM, and Motorola, established a rapidly expanding market for RISC-based hardware and software. IBM has several successful lines of PowerPC-based products for workstations and servers. Motorola has recently introduced a broad range of desktop and server systems, and other companies such as Bull, Canon and FirePower have announced or shipped PowerPC-based systems. Apple has Power Macintosh systems, and companies such as Daystar, Pioneer, Power Computing and Radius also have announced Power Macintosh-compatible systems.

Multimedia, object-oriented technology, and simulations require ever-more-powerful operating environments, and a fast-growing POWER and PowerPC architecture is essential. Efforts are under way to build an industry standard as well as commodity-priced components and subsystems.

3.1 POWER

The reduced instruction-set computer (RISC) architecture traces its heritage to a machine called the IBM 801 minicomputer developed in the mid-'70s. The first implementations of the POWER architecture were based on seven or nine chips, depending on the RS/6000 model. The seven-chip complex had a 32 KB data cache, and the nine-chip complex had a 64 KB data cache. This implementation provided exceptional performance.

The architecture was developed to address the mid-range requirements of the IBM AIX family supporting engineering-scientific and commercial application environments. It allows balanced performance between fixed-point and floating-point and provides a near vector computer performance in numerically intensive applications, while still performing well on scalar codes. It supports an efficient implementation of instruction and data caches. The virtual memory architecture allows efficient mapping of file and shared objects among a large number of concurrently active processes.

These features are achieved by allowing parallel execution of distinct functional units. This enables the execution of as many instructions per cycle as the processor implementation and the compiler technology would allow. Compound function instructions execute in a single cycle in their respective units, but they perform two or more basic RISC instructions.

3.1.1 Functional Units

The POWER RS/6000 is designed based on three independent functional units:

- Branch processor unit
- Fixed-point processor unit
- Floating-point processor unit

Each of these functional units is designed for a maximum concurrence between the units; 184 instructions are divided between the functional units to minimize their interaction and synchronization.

Branch Processor Unit (BPU): The branch processor unit processes an incoming instruction stream from the instruction cache and passes a flow of instructions to the fixed-point and floating-point processor units. It provides the branch, interrupt, and condition code functions within the system. Also, the branch processor unit supports the supervisor call (SVC) instruction, which is a software interrupt. It contains six special registers.

Fixed-Point Processor Unit (FXU): The fixed-point processor unit is designed to execute all 79 fixed-point arithmetic and logical instructions as well as all 55 data reference instructions. It has 32 32-bit general-purpose registers and five special registers.

Floating-Point Processor Unit (FPU): The floating-point processor unit supports the execution of all 21 floating-point instructions. It has 32 64-bit floating-point registers and a floating-point status and control register.

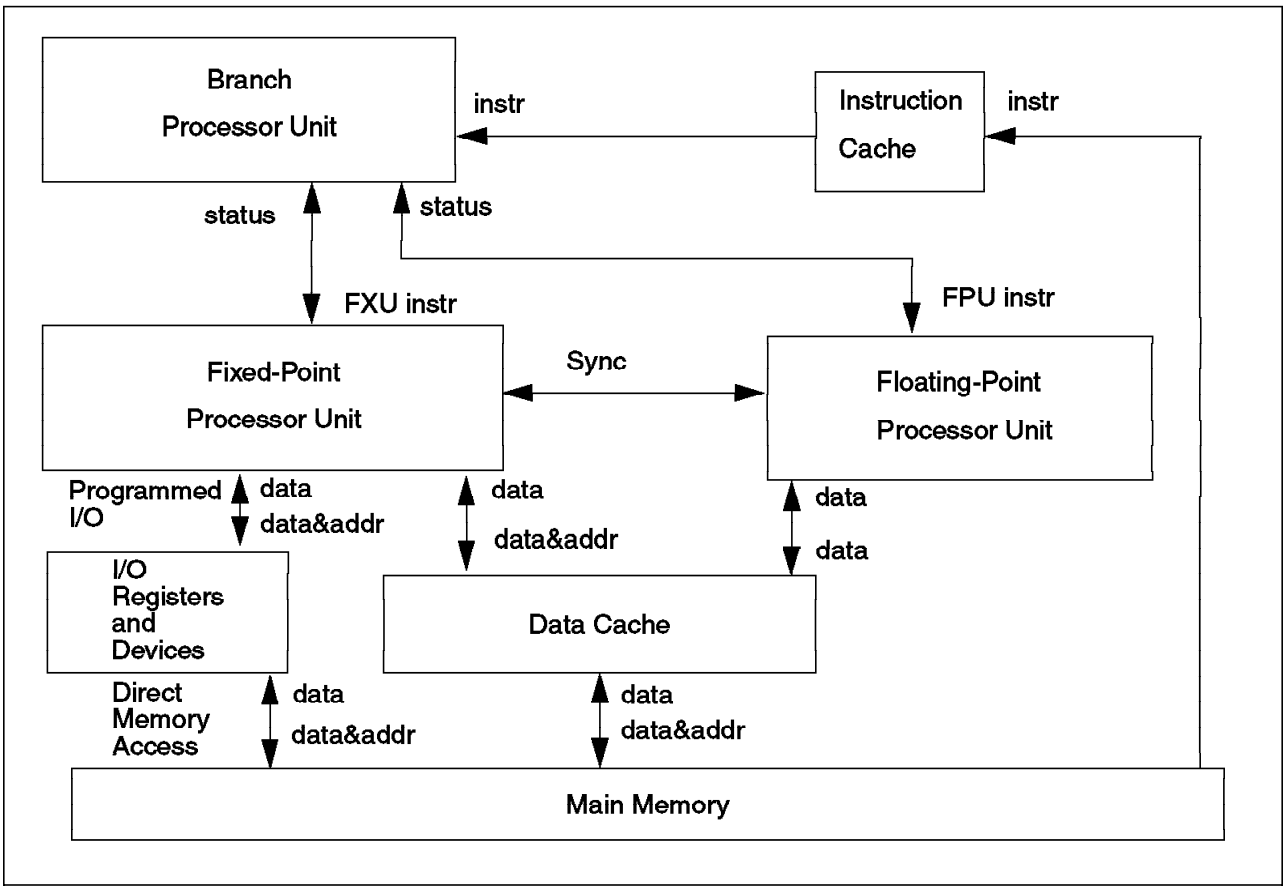


Figure 11. Logical View of POWER Architecture

3.1.2 Storage Control

The virtual memory architecture provides a 4 PB (2^{52}) virtual address space and a 4 GB (2^{32}) real address space made up of 4 KB pages. This architecture contains an 8 KB instruction cache that was expanded to 32 KB in subsequent systems. A hardware locking mechanism provides hardware-assisted locks. The architecture has been enhanced to provide for the management of software-visible instruction and data caches.

Virtual Address Architecture: The instructions and data references generated by programs are all 32-bit effective addresses. The most significant four bits of the effective address are used to select one of 16 segment registers.

Each segment register can be assigned to memory or I/O space. If the most significant bit of the segment register is 1, this request is sent to I/O space along with the content of the segment register. If the bit is 0, the least significant 24-bits of the segment register (the segment ID) are concatenated with the remaining 28-bits of the effective address to create the 52-bit virtual address. To the executing program, memory appears to be 4 GB of virtual memory broken into 16 segments of 256 MB each. Over 16 million (2^{24}) unique segments are available, which can easily support all open files and active objects of a large number of concurrently active processes across one or more processors.

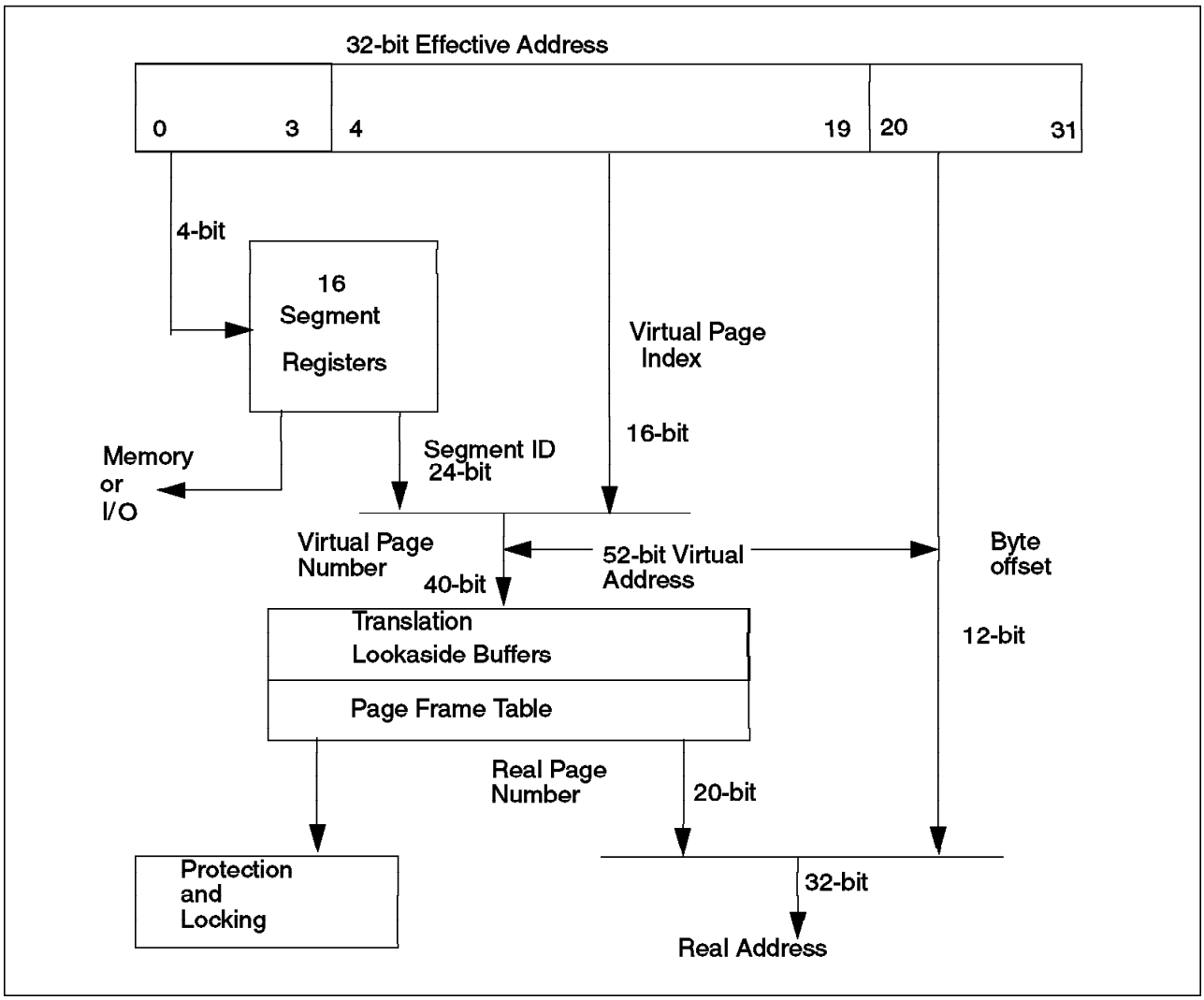


Figure 12. Virtual Address Generation and Translation

To efficiently map these large virtual addresses to their respective real page frames, an inverted page table structure is used. The page frame table (PFT) contains one entry per real page frame.

Each entry in the PFT contains the virtual address to which the real page is currently assigned. A virtual-to-real translation is performed by hashing into a hash anchor table (HAT). Each HAT entry is pointing to the PFT, where a search for a matching virtual address continues until an invalid pointer is found, at which point, a storage interrupt is taken.

The most recent translations are maintained in translation lookaside buffers (TLB) in the processor to avoid repeating these searches.

Cache Architecture: The POWER implementation makes the instruction and data cache explicitly visible to the software. This increases the parallelism between the branch and the fixed-point processor unit as well as between I/O devices and these units.

The design of the data cache unit (DCU) provides a 64 KB, four-way, set associative, store-back cache memory divided into four identical chips. It has

separate interfaces of 4 bytes to the FXU, 8 bytes to the FPU, and 8 bytes to the I/O control unit.

This design, with its large store-back cache and its memory and I/O buses, gives the processor complex the ability to quickly transfer large amounts of data to and from the memory and I/O. Faster and denser memory cards have been added, and new devices and buses have been designed for the systems without affecting the processor complex. These features have boosted processor speed and kept the CPU from having to wait continuously for data from memory or I/O.

3.1.3 Instruction Set

The POWER architecture instruction set is defined with as much function per instruction as possible, revealing all of the parallelism in the machine to the compilers, which can fully exploit the capability of multiple operation cycles.

The architecture defines 184 instructions and allows multiple execution units for fixed-point, floating-point, and branch processing, focusing on reduced instruction-set cycles.

One of the most important instructions is the floating-point multiply-add (FMA) instruction. The FMA compound instruction consists of a floating-point multiply and a dependent add. It performs a multiplication of two registers and adds a third register to this intermediate result with a total latency of only two cycles. Independent FMA instructions can start every cycle, allowing a peak number of floating-point operations at twice the MHz rate, while using a single functional unit.

The design includes multiple sets of condition codes that are scheduled and managed by the compiler much like the general-purpose registers are scheduled and managed for load and store instructions. This allows several condition registers to alter operations to be processed at the same time, and it increases parallelism between fixed-point and floating-point units.

Many commercial applications use string-move operations. Therefore, instructions with hardware assistance were added to move strings at rates approaching cache bandwidth.

3.1.4 Performance

The performance results achieved with the compiler technology and the POWER implementation were a good starting point for the future development of the POWER microprocessor family. Compared to CICS contemporary microprocessors, the POWER architecture has reduced the number of cycles on the instruction set. Analysis of compiled code for various commercial and scientific applications has shown a path length for the instruction set that is less than or equal to those of CICS processors.

The ability to execute floating-point operations at the same speed as integer operations is one goal. It makes a floating-point unit capable of completing two floating-point operations every clock cycle. At a clock cycle of 25 to 30 MHz this translates to a sustainable peak of 50 to 60 million floating-point operations per second. This assumes that sufficient cache bandwidth and fixed-point and floating-point concurrence exists to sustain the maximum hardware rates.

Performance is enhanced via several instructions that often are considered more complex than the traditional RISC definition. These include the floating-point

multiply-add (FMA) instruction, a branch-on-count (BCT) operation, and update forms of storage references. Many experts credit the FMA instruction as a key component of the RS/6000 processor's outstanding floating-point performance.

Combining the single-cycle nature of the floating-point unit with overlapping fixed-point and branch operations and independent cache paths of one word for fixed-point data and two words for floating-point data, performance of this implementation approaches that of large traditional mainframes and the early generations of supercomputers.

3.2 RSC (RISC Single Chip)

Advances in silicon technology led to the RISC single chip, a second POWER processor design that included more than a million transistors on one chip.

This design integrated a simple branch unit, fixed-point unit, floating-point unit, unified cache, memory controller, and an I/O controller on a single chip for use in low-cost desktop systems. The capability of this design led to its adoption as the starting point for the next step in the POWER family, the PowerPC microprocessors. The RSC supports the same instruction set as POWER and provides binary compatibility for all AIX applications.

3.3 POWER2 Multichip

The early POWER2 processor complex consists of eight semi-custom chips partitioned in the same way as the POWER: the instruction cache unit (ICU) which also processes branches, the FXU, FPU, four data cache units (DCUs), and a storage control unit (SCU). The early models of the POWER2 were announced in September 1993 and are the basis of the POWER2 Super Chip (P2SC) version. The POWER2 architecture expands the cache capacity, doubles the number of functional units, doubles the bandwidth of most buses, and quadruples the cache to floating-point bandwidth.

The POWER2 architecture offers numerous performance gains:

- New instructions enhance floating-point bandwidth, square root, and data conversion.
- User-level instruction set is a superset of the original POWER set. Existing POWER binaries can run unmodified on the faster systems.
- Existing POWER binaries gain the performance benefits of the larger data cache, higher clock rates, improved bandwidth, and additional functional units.
- Recompiling can maintain portability while providing gains from added compiler transformations. When portability is not required, a recompiled application can also exploit new instructions.

POWER implements one floating-point, one integer, and one branch unit. It can execute four instructions per cycle (five operations).The POWER2 multichip implements dual integer, floating-point, and branch units that can execute six instructions per cycle (eight operations).

The architecture features high-performance, floating-point storage access instructions, load quad word (128 bits) and store quad word, which support all of the addressing forms of double-precision storage references. The load quad

word moves two adjacent double-precision storage operands into two adjacent floating-point registers.

The instruction cache size is 32 KB, two-way set associative, and the data cache is either 128 KB or 256 KB, four-way set associative single line, depending on the RS/6000 model.

The following are some other differences in the POWER2 multichip compared to the POWER architecture:

- Eleven new instructions (giving a total of 195 instructions)
- Different page frame table format (hash instead of inverted page frame tables)
- Page aliasing
- Support for floating-point imprecise mode
- Different interrupt mechanism
- Faster clock speed
- Double the bandwidth from memory to the cache (eight-word bus, 128 bytes)
- Double bandwidth from the cache to the integer (two single words,16 bit) and floating-point units (two quad words, 32 bit)
- Different alignment requirements for the quad word data
- A special data address break-point register
- A performance monitoring facility

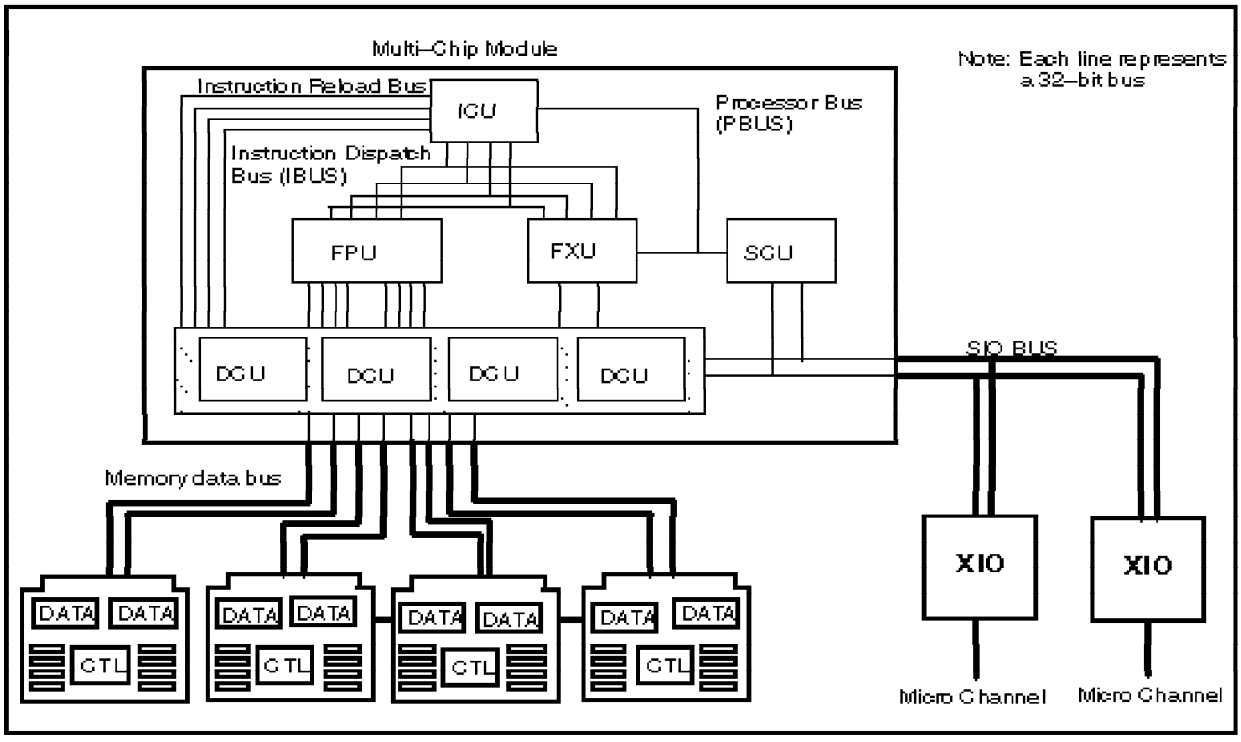


Figure 13. POWER2 Eight-Word System

3.3.1 POWER2 Super Chip

The POWER2 Super Chip (P2SC) is a compression of the POWER2 eight-chip architecture into a single chip with increased processor speed and performance. It retains the design of its predecessor, the POWER2.

The initial models have clock speeds of 120 MHz and 135 MHz. High-density CMOS-6S technology allows each to incorporate 15 million transistors.

The most significant change is a halving of the size of the data cache and the data TLB, which now are 128 KB and 256 KB, respectively. These changes were required to fit the eight-chip processor onto a single chip.

The P2SC delivers the processing and dual floating-point power needed for large, numeric-intensive tasks as well as the integer and transaction performance for commercial applications. The P2SC contains on-chip 32 KB instruction and 128 KB data cache and is full binary compatible with the POWER2 architecture.

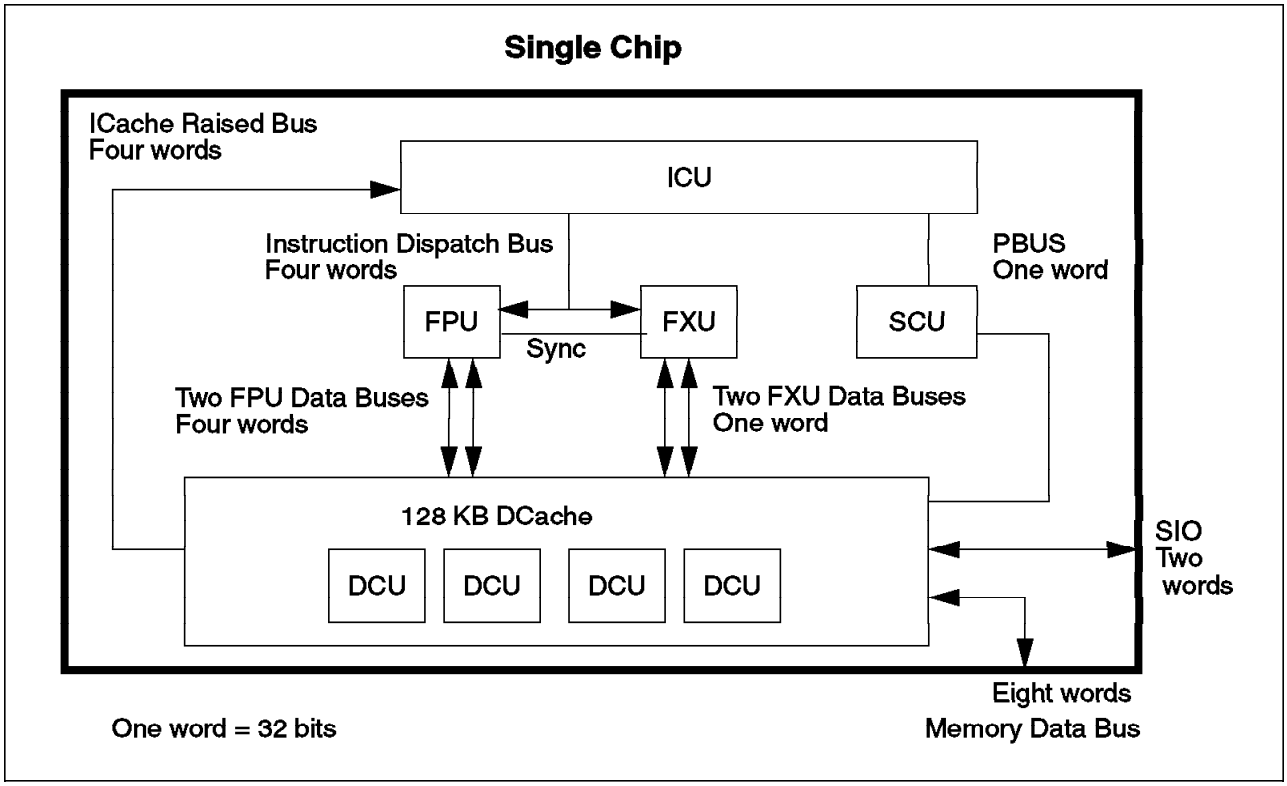


Figure 14. POWER2 Super Chip Module

The P2SC can issue and execute six instructions per cycle, two of which can be floating-point multiply-add (FMA) instructions. It supports register renaming and out-of-order execution, although only for floating-point instructions. With dual branch units, the P2SC can also execute two branches per cycle, although only one can be taken, while the other is put on hold.

The data cache is triple-clocked to handle two CPU accesses (load or store) plus a cache refill (write to cache from main memory). Thus, this part of the chip operates at 500 MHz.

The P2SC's great performance is directly associated with the inclusion of up to 2 GB of DRAM across a 256-bit-wide interface. This interface plugs the processor directly into a byte stream bus of up to 2.2 GB per second. The chip also integrates a 64-bit I/O bus for peripheral interconnection.

3.4 PowerPC

The PowerPC architecture uses the POWER design as a base, exploiting its single-chip capabilities and reducing the system's cost. The ability to maintain compatibility across a broad spectrum of single-chip and multichip implementations is part of IBM's ″Palmtops to Teraflops″ strategy.

The PowerPC architecture was developed focusing on changes in the existing POWER architecture that would allow more aggressive superscalar implementations, allow higher clock rates and require less silicon. The IBM, Apple, and Motorola alliance defined some goals for the architecture:

- Make it simple enough to allow a processor design that has a very short cycle time.
- Minimize effects that hinder the design of aggressive superscalar implementations.
- Enable a broad range of implementations, from low-cost controllers to high-performance processors.
- Include multiprocessor features.
- Define a 64-bit architecture that is a superset of the 32-bit architecture, providing binary compatibility for 32-bit applications.

The flexibility of the architecture allows developers to produce different PowerPC chips that vary in complexity, speed, power consumption, number of execution units, cache sizes, multiprocessor capability, and so on, without affecting the binary compatibility of the applications that will run on them. The architecture also provides a stable instruction set for applications and operating systems.

The PowerPC includes most POWER instructions. The architecture group identified some features of the POWER architecture that were restrictive, too complex to implement efficiently, or not cost-effective. This resulted in excluding 32 instructions from the instruction set. The excluded POWER instructions are executed infrequently and can be replaced with several instructions that are used in both architectures. The excluded instructions will cause a program interrupt on the PowerPC processors and will be emulated by the AIX operating system. Also, PowerPC defines new instructions and features to complete the architecture and achieve the goals set by the architecture group.

Where possible, the architects defined additional features to avoid adding hardware complexity to the PowerPC microprocessor. A full set of single-precision floating-point arithmetic instructions were added, plus two instructions to convert floating-point values to integers. Because most of these instructions provide the single-precision version of existing double-precision instructions, complexity was not much greater than adding hardware to the design.

The different implementations of the PowerPC will be described in the following sections.

3.4.1 PowerPC 601

The PowerPC 601 microprocessor was the first PowerPC chip. It established the PowerPC architecture in the market, with the compromise to offer a competitive microprocessor at a low cost and to be suitable for a wide range of system designs, including those requiring support for symmetric multiprocessing. The RISC single chip (RSC) became the base design for the PowerPC 601.

The PowerPC 601 provides a versatile system interface that allows for a wide range of implementations. The interface includes a 32-bit address bus, a 64-bit data bus, and 52 control and information signals.

The PowerPC 601 integrates three executions units: a fixed-point unit (FXU), a branch processing unit (BPU), and a floating-point unit (FPU). As a superscalar processor, it is capable of issuing and dispatching three instructions per cycle, one to each of the execution units in parallel. Most integer instructions execute in one clock cycle, and the FPU is pipelined so a single-precision multiply-add (FMA) can be issued every clock cycle.

The 601 provides 32-bit effective (logical) addresses, integer data types of 8, 16, and 32 bits, and floating-point data types of 32 and 64 bits. Included on-chip is a 32 KB, eight-way set associative, physically addressed, unified instruction and data cache, and an on-chip memory management unit (MMU).

The MMU contains a 256-entry, two-way set associative, unified translation lookaside buffer (UTLB) and provides support for demand-paged virtual memory address translation and variable-sized block translation.

Additional on-chip snooping logic maintains cache coherency in multiprocessor applications. The 601 supports single beat and burst data transfer for memory accesses. It also supports both memory-mapped I/O and I/O controller interface addressing.

The chip I/Os include a 32-bit address bus and a 64-bit data bus. In addition, the chip supports a special asynchronous serial port, the common on-chip processor (COP) bus, to provide advanced debugging and testing features.

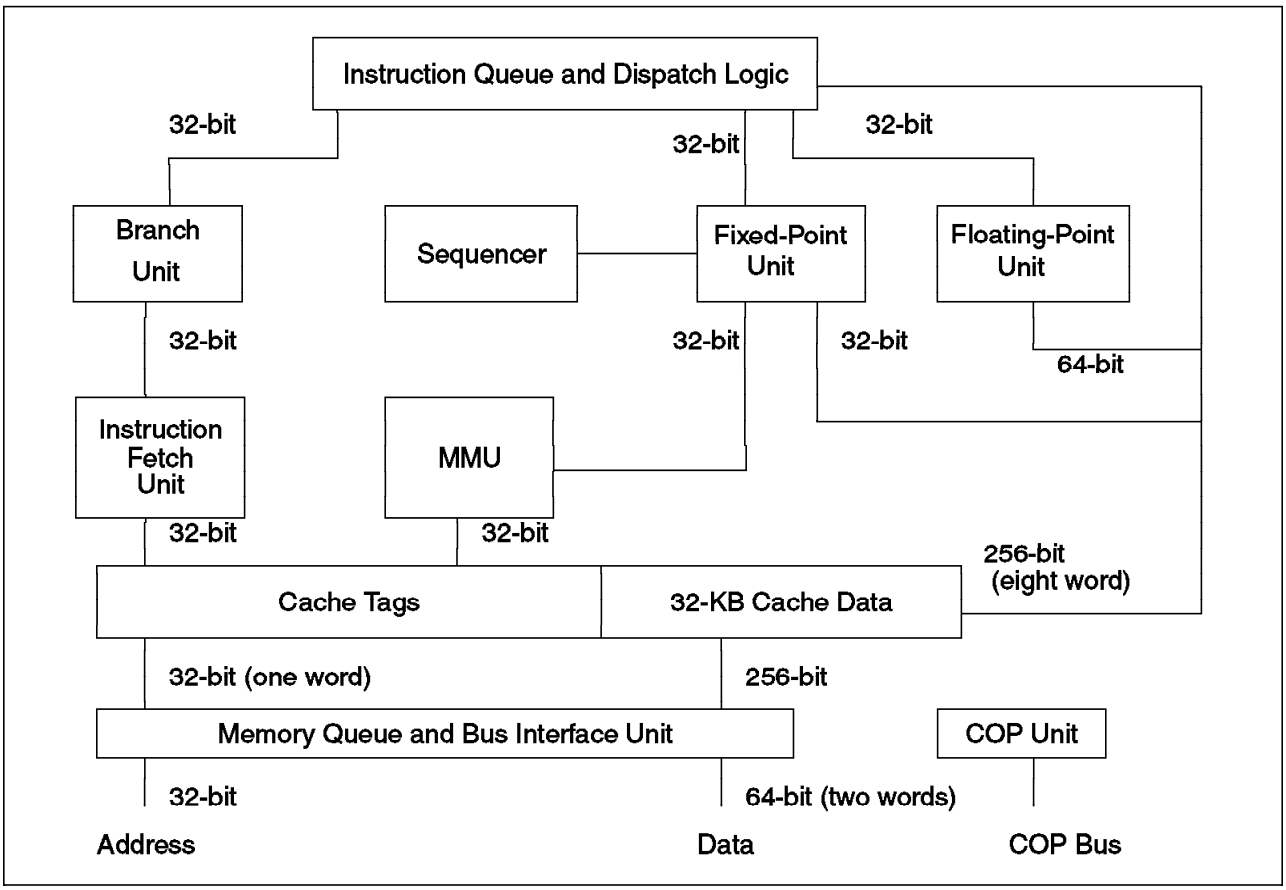


Figure 15. The PowerPC 601 Microprocessor

The PowerPC 601 architecture contains several functional units, as shown in Figure 15. Some units perform the same tasks as the POWER architecture does, but some units are new for this architecture. The following sections briefly describe each of these units.

Instruction Queue and Dispatch Unit (IQDU): This unit provides centralized control of the instruction flow to the execution units. The instruction unit determines the address of the next instruction to be fetched based on information from the sequencer and the branch unit. The instruction unit also enforces pipeline interlocks and controls feed-forwarding.

Instruction Fetch Unit (IFU): The IFU coordinates instruction fetching from the cache. Several units generate instruction fetch addresses. The BPU provides addresses that result from branch instructions. The sequencer unit provides addresses associated with interrupts and other synchronizing events. The instruction fetch itself generates the next sequential address if no branches or interrupts have occurred. During each cycle, the processor selects, translates, and forwards the appropriate address to the cache arbitration logic for considerations to access the cache. The IQDU accepts and processes instructions fetched from the cache.

Branch Processing Unit: The branch unit performs the same tasks as in the POWER architecture. In addition, the 601 BPU uses a bit in the instruction encoding to predict the direction of the conditional branch. The BPU dispatches and completely executes unconditional branches (or branches that were unresolved) in a single cycle. The branch instruction pipeline has only two

stages (fetch and dispatch-decode-execute-predict) that allow the processor to react quickly to branches detected in the instruction stream and to reduce the latency of subsequent instructions.

Fixed-Point Unit: Like the POWER FXU, the PowerPC FXU floating-point memory accesses instructions in agreement with the FPU. The instructions flow through a four-stage pipeline (fetch, dispatch-decode, execute, write-back), and the hardware automatically handles all exceptions.

Floating-Point Unit: The FPU features both single-precision and double-precision floating-point operations. It also supports the compound multiply-add (FMA) operations that are defined for the PowerPC architecture. The FPU pipeline consists of six stages (fetch, dispatch, decode, multiply, add, write-back).

Memory Management Unit: The MMU performs the virtual-to-real address translation for load and store instructions. The MMU also controls access privileges for the physical memory and virtual memory on block and page granularity. Referenced and changed statuses are maintained by the processor for each page to assist implementation of a demand-paged virtual memory system.

Cache Structure: The 601 microprocessor provides a 32 KB, eight-way set associative, unified cache. The line size is 64 bytes; each is split into 32-byte sectors. The cache is indexed with a real (or physical) address, and the tags are associated with the real address, as well. This structure uses several techniques to achieve the most effective use of the bandwidth. For example, it has an eight-word read interface, which at 66 MHz, allows an effective 2.1 GB/s data rate. This bandwidth is especially important for instruction fetching, snoop pushes, and multiprocessor cast-outs. The cache can perform a complete read-modify-write (a store operation) in a single cycle.

The designers developed a balanced arbitration scheme to prioritize the various cache access requests that can occur each cycle. To minimize the effect that the cache arbitration stalls have on the execution units, queues for fixed-point and floating-point storage operations are located above the cache.

Memory Queue and Bus Interface Unit: The bus interface unit (BIU) converts operations in the memory queue into transactions on the 601 bus. The interface defines a 32-bit address bus and a 64-bit data bus. These buses are logically and physically decoupled from one another so that the protocol can support system bus organizations that use pipelined, non-pipelined, or split bus transactions. To reduce latency and increase performance, the 601 itself can pipeline up to two operations onto the bus.

The memory queue cooperates with the cache and the BIU to effectively handle operations that require access to and from the 601 bus interface. It also allows the processor to decouple the cache from the bus interface, which improves the overall effectiveness of the cache.

Sequencer Unit: The sequencer unit is an embedded support processor that assists the core CPU in handling many algorithmic and area-intensive functions of the PowerPC architecture. The unit sequences the power-on reset functions that include an array self-test for the cache and array initialization for all other storage cells on the chip. It maintains an on-chip real-time clock (RTC), and it

handles errors, control of context-synchronizing events, and the recording and sequencing of interrupts.

Common On-Chip Processor Unit: The COP is the master control logic for the built-in self-test (BIST), the debug, and the test features of the 601 chip. A simple serial command interface allows external devices to communicate with the COP and initiate various actions.

3.4.2 PowerPC 603 and 603e

The PowerPC 603 is the first implementation of the PowerPC especially suitable for low-consumption computers. The 603 provides both industry-leading RISC performance and power management for the notebook and energy-sensitive desktop market.

The memory management has 52-bit virtual and 32-bit real addressing, with a set associative instruction cache of either two-way 8 KB or four-way 16 KB, and the same configuration for the data cache. The bus interface contains 32-bit address and a 32- or 64-bit data bus.

High performance is achieved through concurrent execution. The 603 is a superscalar processor capable of issuing and dispatching as many as three instructions in five parallel execution units. The instructions can execute out of order for increased performance, however, the 603 makes completion appear sequential.

Fixed-Point Unit: The 603 FXU adds, subtracts, shifts, or rotates in one cycle. It executes hardware multiplication and division with 32 32-bit general-purpose registers (GPR).

Floating-Point Unit: The 603 FPU is optimized for single-precision multiply-add FMA, and executes single-precision and double-precision floating-point arithmetic with 32 32-bit floating-point registers (FPR).

Branch Unit: The BPU branches instructions from the fetch unit with the same performance as the other PowerPC architectures.

System Unit: The SU executes condition register logic, special register transfer, integer add/compare instructions, and other system instructions.

Load/Store Unit: The 603 LSU executes all load and store instructions and provides the data transfer between the GPRs and FPRs and the cache/memory subsystem.

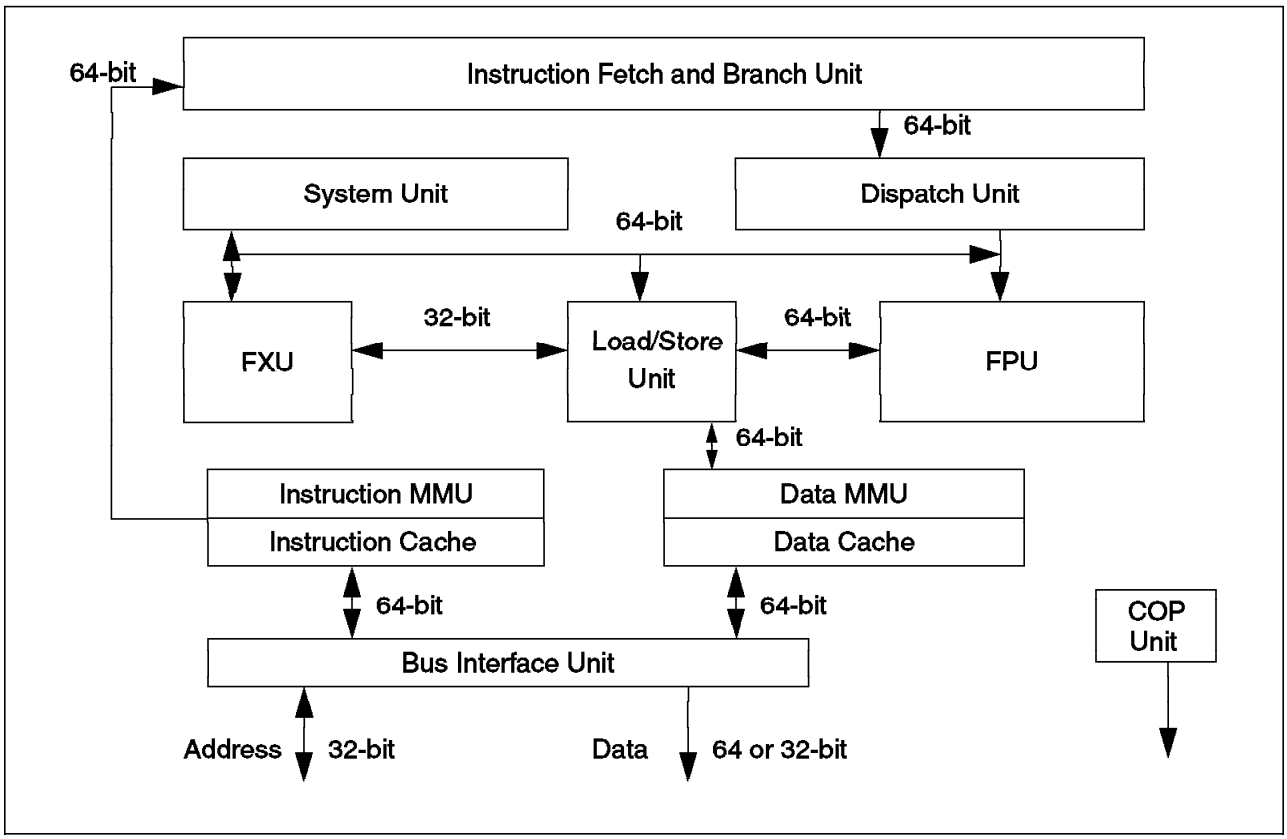


Figure 16. The PowerPC 603 Microprocessor

The 603e enhancements include:

- Higher clock frequencies delivering higher levels of performance.
- A performance-enhancing feature supporting misaligned little endian accesses.

3.4.3 PowerPC 604 and 604e

The PowerPC 604 microprocessor family is a 32-bit implementation of the PowerPC architecture that is software and bus compatible with the PowerPC 601 and PowerPC 603 microprocessors. The 604 gives the performance needed to support graphics-, computation-, and multimedia-intensive applications. The early 604 implementation reaches new levels of performance by issuing four instructions per cycle, thus achieving balanced execution of integer and floating-point operations.

The 604 uses a superscalar design to provide six independent execution units: one branch unit, three fixed-point units, one floating-point unit, and a load/store unit. The 604 chip also uses dynamic branch prediction techniques to enhance instruction pre-fetching as well as speculative execution techniques to take advantage of the improved instruction pre-fetching and multiple execution units.

On-chip, the 604 features 16 KB instruction and 16 KB data caches coupled to a high-performance, 64-bit system bus. The microprocessor takes advantage of instruction-level parallelism found in today's application programs.

The 604 fetches, dispatches, and completes up to four instructions per cycle. It can hold up to eight instructions for dispatch and 16 more in various stages of

execution. From the six execution units, three are pipelined, to sustain a four-instructions-per-cycle rate for those applications that offer a high degree of parallelism, and a total of six pipeline stages are used to achieve its 100 MHz initial design. The stages are: fetch, decode, dispatch, execute, complete, and write-back.

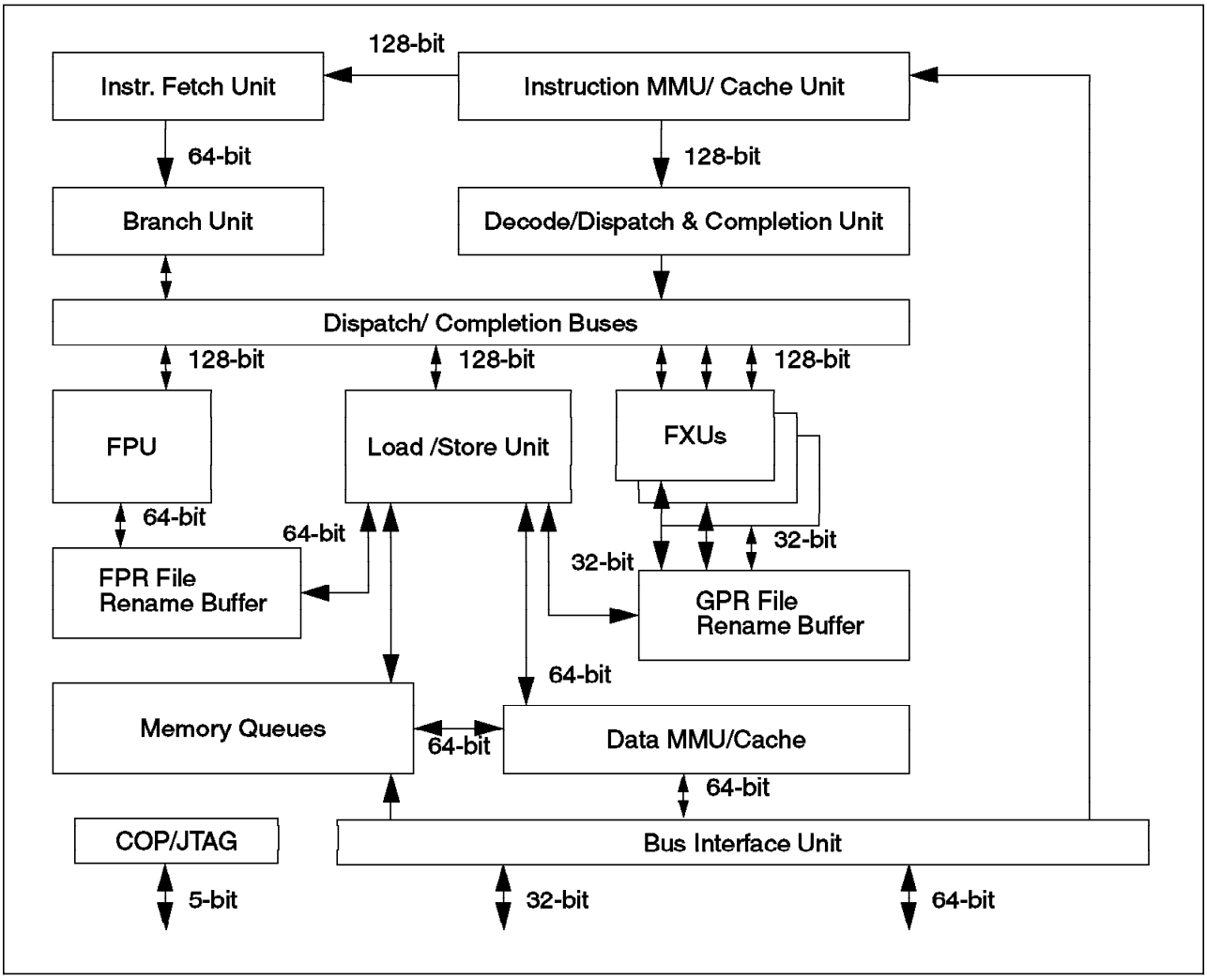


Figure 17. The PowerPC 604 and 604e Microprocessor

The 604e is targeted at the workstation, PC server and power user desktop segment. Enhancements to the PowerPC 604e include:

- Size of instruction and data cache have doubled.
- Higher clock frequencies.
- A built-in performance monitor.

3.4.4 PowerPC 620

The PowerPC 620 is the first 64-bit implementation of the PowerPC architecture supporting 32- and 64-bit applications. The 620 is specified for 64-bit addressing, which provides 64-bit effective (logical) addresses, integer data types of 8, 16, 32, and 64 bits, and floating-point data types of 32 and 64 bits (single-precision and double-precision).

The 620 is a superscalar processor capable of executing four instructions per cycle. It can actually issue four instructions simultaneously, and as many as six instructions can finish execution in parallel. This parallel design allows rapid execution times and yields high efficiency and throughput.

The microprocessor has six execution units that can operate in parallel:

- Floating-point unit (FPU)
- Branch processing unit (BPU)
- Load/store unit (LSU)
- Three integer units (IUs):
  - Two single-cycle integer units (SCIUs)
  - One multiple-cycle integer unit (MCIU)

Branch prediction, instruction pre-fetching and speculative execution are used to take advantage of multiple execution units. Instructions are dispatched to the execution units in program order, executed out-of-order, and completed in-order to support precise exceptions.

The 620 has separate memory management units (MMUs) and separate 32-KB, eight-way set associative, on-chip caches for instruction and data, a 128-entry, two-way set associative translation lookaside buffer (TLB) for instructions and data, and support for demand-paged virtual memory address translation and variable-sized block translation.

The 620 provides an integrated L2 cache controller that supports L2 configuration from 1 MB to 128 MB, using the same block size of 64 KB per line as the internal L1 caches. The L2 cache is a direct-mapped, error correction code (ECC) protected, unified instruction and data secondary cache that supports the use of single- and double-register synchronous static RAMs.

The PowerPC 620 has a 40-bit address bus and can be configured with either a 64- or 128-bit data bus.

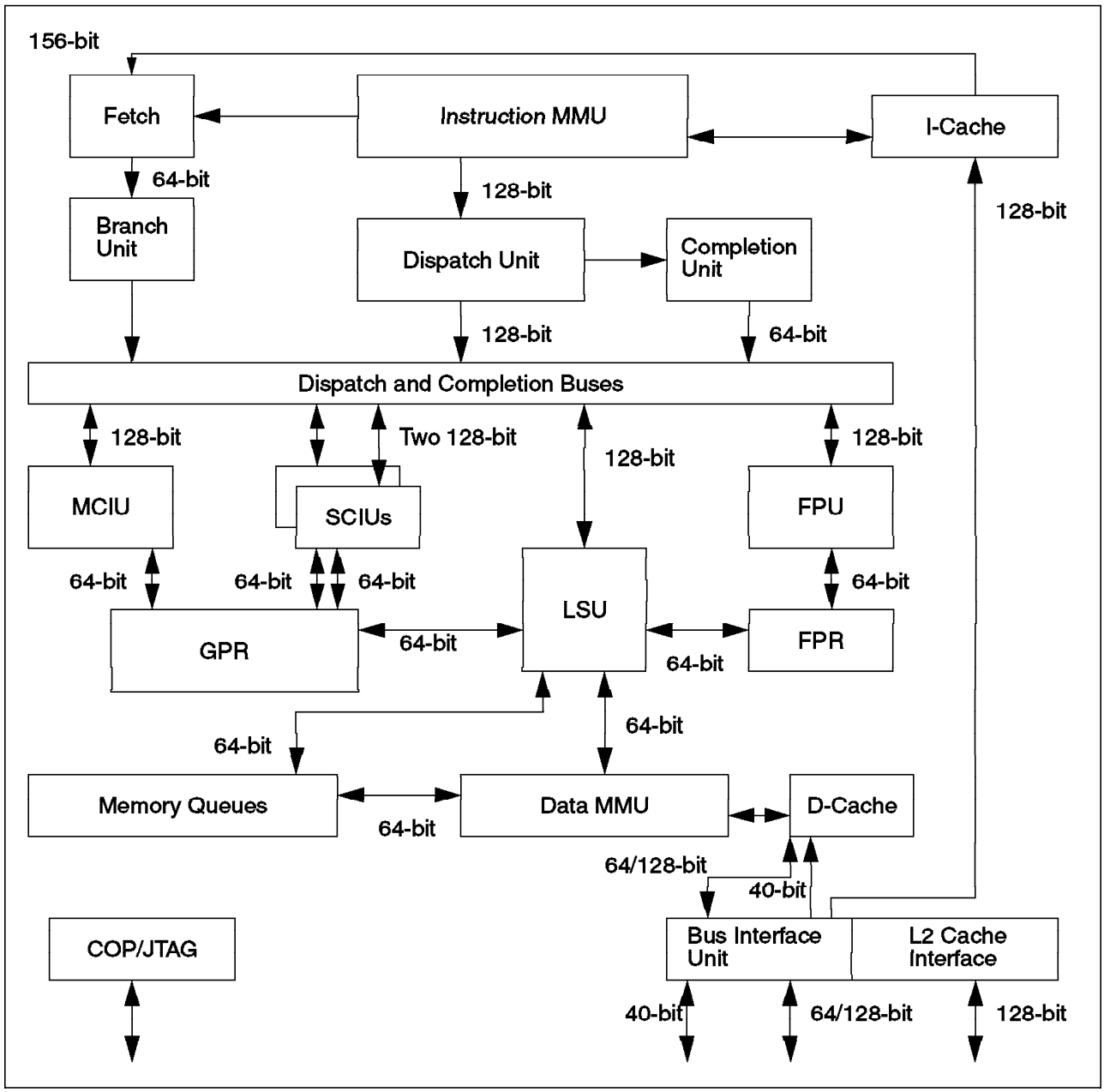


Figure 18. The PowerPC 620 Microprocessor

The 620's MMU and exception model support demand-paged virtual memory. Virtual memory management permits execution of programs larger than the size of physical memory. Demand-page implies that individual pages are loaded into physical memory when they are first accessed by an executing program. The MMU supports up to one heptabyte (2^{80}) of virtual memory and one terabyte (2^{40}) of physical memory. It also supports block address translations, direct store segments, and page translation of memory segments.

3.5 Conclusion

The POWER, POWER2, and PowerPC architectures are high-performance, superscalar implementations. The POWER2 and PowerPC allow optimizing compilers to schedule instructions to maximize performance through efficient use of the instruction set and register model. The multiple independent execution units allow compilers to maximize parallelism and instruction throughput. Compilers that take advantage of the flexibility of the architectures can additionally optimize system performance of the processors.

3.6 References

http://www.rs6000.ibm.com/resource/technology

http://fnctsrv0.chips.ibm.com

http://www.developer.ibm.com/library/aixpert/feb94/feb94toc.html

IBM RISC System/6000 Technology, SA23-2619

PowerPC and POWER2, Technical Aspects of the New IBM RISC System/6000, SA23-2737

/AIXtra, September/October 1994

IBM Journal of Research & Development, September 1994

Microprocessor Report, Microdesign Resources, August 26, 1996

PowerPC 601, RISC Microprocessor User's Manual, SA14-2007

PowerPC 603e. RISC Microprocessor User's Manual, SA14-2029

RISC System/6000 Hardware Quick Reference Guide

Chapter 4. Hardware

This chapter provides relevant information on processor and memory features, as well as hardware products such as disk storage, asynchronous communication adapters, LAN/WAN adapters, and graphics adapters.

4.1 Processor

Efficient pipelining of instructions and data allows the RS/6000 to provide exceptional performance. However, this performance is heavily influenced by the type of application being measured and the actual design of the code being executed. Applications that run primarily in cache such as the LINPACK benchmarks will yield results comparable to those of the synthetic benchmarks. Applications that perform multiple data accesses or work with code that is not tightly looped may not yield the expected performance.

Furthermore, processors are fed by relatively slow memory. The performance of processors increases about twice every two years. Today's typical processor cycles are about 6 ns - 3 ns (166 MHz - 333 MHz), whereas in 1986, processor cycles were about 120 ns (8 MHz). Meanwhile, the memory access time has only dropped from 120 ns to 60 ns. For example, a processor running at 333 MHz (3 ns) will have to wait at least 20 cycles to access a data in the main memory. Based on this, considering only the processor's clock speed as a performance indicator can lead to meaningless results. Today's processors can spend a high percentage of time just waiting for information. The faster the processor, the more it will have to wait for data from the main memory. As an example, some processors running in commercial environments can spend 10 to 50 percent of their time stalled, waiting for instructions or data. This idle time is not reported by the system (`vmstat`, `sar`) as the system thinks the processor is busy. This shows that the memory subsystem (caches, buses, bandwidth and latency) design is a key point for today's computer performance.

4.1.1 Cycles Per Instruction

Cycles per instruction (CPI) and instructions per cycle (IPC=1/CPI) are common measures of processor efficiency.

The infinite cache CPI is a gauge that gives the relative efficiency of a processor on a specific workload. Its value depends on the workload as well as on the processor itself. The main advantage of using CPI is that it is additive with other CPI components. Depending on the miss rate, each component (L1, L2 and memory) will add its number of CPI. This helps in estimating the overall number of CPI for a given workload and, thus, in estimating the power of the system for that workload.

For example, PowerPC processors are able to execute 4 instructions in one cycle (4 IPC = 0.25 CPI).

The PowerPC 604+ 166 MHz measures about 1.12 CPI (min 0.8 and max 1.75) when running the SPECint95 benchmark. In other words, for this benchmark, the processor executes an average of 0.9 instructions per cycle.

This shows that, depending on the application you are running, there is a large difference between the theoretical performance of your processor and the reality.

4.2 Memory

When considering the performance of a system, the processor and the memory should not be taken as separate devices but as tightly coupled elements that closely interact.

Here are some hardware system parameters that can influence performance:

- Frequency ratio between the processor speed and the L2 cache (e.g. 2:1 for a processor speed of 100 MHz and an L2 bus speed of 50 MHz)
- L1 cache size, line size, associativity
- L2 cache size, line size, latency and intervention
- Cache replacement algorithms
- Bus speed, width and protocol
- Frequency ratio between the processor speed and the memory bus speed
- Memory subsystem latency

Intervention

This operation happens when a processor wants the data already in a cache of another processor. Cache-to-cache copying is usually done without using the main memory to increase the transfer speed.

4.2.1 Memory Hierarchy

Since having caches improves data locality, most systems implement a multilevel cache structure.

In Chapter 2, “Background” on page 5, we introduced the L1 and L2 cache concept. For example for the POWER2 RS/6000, typical L1 cache lines are 128 or 256 bytes long. A 32 KB cache holds 256 or 128 lines, depending on the line size. L2 cache may be as large as 1 MB or more.

The RS/6000 also uses the concept of the translation lookaside buffer (TLB) to improve memory performance. Programs use addresses that must be translated into real memory addresses. To quickly translate addresses, the TLB holds 512 entries (two banks with 256 each) of recently used translations for virtual to real addresses.

Caches and TLB are supported by hardware. Hence, they can quickly translate the virtual addresses into physical locations of data. Paging-related activities are managed by the virtual memory manager and take more cycles to accomplish. The page frame table (PFT) holds as many entries as the number of pages of physical memory.

Some memory architectures implement a level 3 cache. The apparent performance benefit must be weighed against the overhead of more and more processing to manage changes to the data. To be efficient, L3 cache should be significantly larger than L2 cache. Generally, L3 cache is implemented on computers that have small L1 cache (8 KB) and small L2 cache (100 KB). L3 cache is usually much more efficient for applications that have a high miss rate.

4.2.2 Memory Cycles

In order to understand the importance of memory hierarchy's performance, here are the number of memory cycles required for a typical processor to access data from the different memory components.

Let us compare the different latencies of the different memory components in a typical system equipped with a processor running at a clock rate in the range of about 100 MHz. The latency is the time it takes to access data from the memory component.

On a typical implementation, it will take one cycle to access data from L1 if there is a cache hit in L1. It will take between seven to 10 cycles to access data from L2 in case of a cache miss in L1 and a cache hit in L2. It will take between 20 to 50 cycles to get data from memory in case of a cache miss in L2. And finally, if you need to access data from disk, it will take between 750000 to 1.5 million cycles. To highlight this point, if we assume that one cycle is one second, it will take 17 days, 8 hours and 40 minutes to access data from disk, whereas accessing data in L1 cache would take one second!

> **Detailed values are hardware dependent. These numbers should only be used as a guide.**

Tuning small user programs often involves cache/TLB management. Tuning large
application programs often involves page and I/O management.

Figure 19 shows the relative access times of different memory levels.

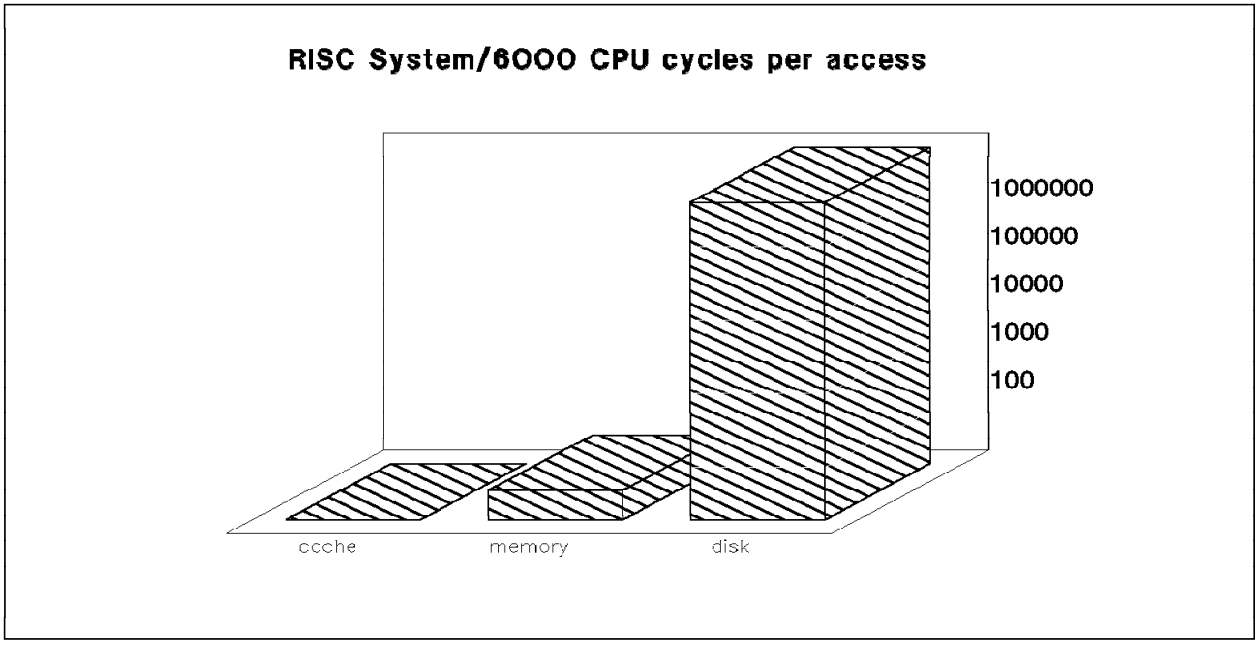


Figure 19. CPU Cycles per Memory Access

Clearly, in terms of performance, accessing data from disk must be avoided at all costs. Thus, a commercial system will have to be designed in such a way

that misses to disks are avoided as much as possible. Since the memory latency is much better than the disk's latency, the memory itself should be used as a huge cache. Therefore, there is a need for high memory capacity.

4.2.3 Scientific, Commercial and System Environment

When studying the performance of a system, it is important to understand the differences between a commercial environment, a scientific environment and the operating system environment.

Scientific Workload Characteristics

- Loops
- Small memory footprint
- Extensive floating-point
- Low OS overhead
- Sequential I/O
- Speed

In a typical engineering/scientific environment, programs are often made of short loops working with data organized by the compiler so they fit into the first level of cache (L1). In such an environment, the L1 hit ratio is very good and usually around 97 percent. This means that 97 percent of the time, data will be found in L1 and 3 percent of the time, the processor will have to fetch data out of L2 (if there is an L2 cache) or from main memory. We will see later that the use of an L2 cache does not increase the program speed-up much.

Comments

Some scientific applications might experience a higher L1 miss rate. Of course, in that case, adding an L2 cache will benefit performance.

Commercial Workload Characteristics

- Branches
- Large memory footprint
- Extensive integer
- High OS overhead
- Throughput

In a transactional database environment, the hit ratio is not as good. Large programs, frequent branches, widely dispersed data references, large numbers of users, large numbers of processes and high process switch rates all combine to produce a high miss rate for the L1 cache. Typically, in such an environment, the hit ratio for L1 is around 85 percent. This means that 15 percent of the time, the processor will have to fetch data from L2 (if L2 exists) or from main memory. We will see that, in this case, an L2 cache helps to improve the performance of the system because L2 cache latency is lower than the memory latency.

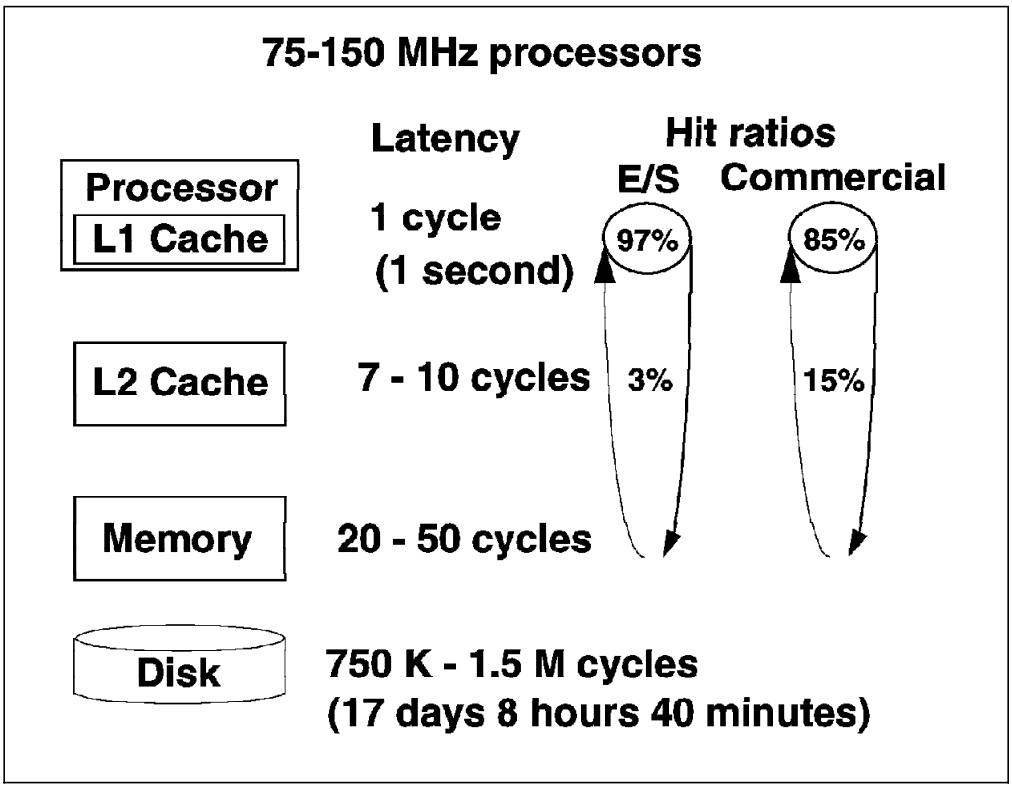


Figure 20. Scientific vs. Commercial Environment

Figure 20 illustrates the memory hierarchy and the differences between a commercial and a scientific environment in terms of hit ratios.

Operating System Workload Characteristics: Scientific applications use the operating system lightly, while commercial applications use the kernel more. Kernel instruction and data cache miss rates are usually higher than commercial user application miss rates. Although kernel code and data structures may be smaller than application code/data structures, the random branching characteristics of the kernel result in a larger footprint. It means that system code generates high cache miss rates and that a smaller amount of system code does not automatically mean a lower overall system miss rate.

If your application -- for example an NFS server -- uses kernel code extensively, you will experience high miss rates for L1 cache, decreasing the overall performance of your system.

4.2.4 Uniprocessor vs. Symmetric Multiprocessor Memory Cycles

The number of memory cycles needed to access data depends on whether the machine is a uniprocessor or a symmetric multiprocessor.

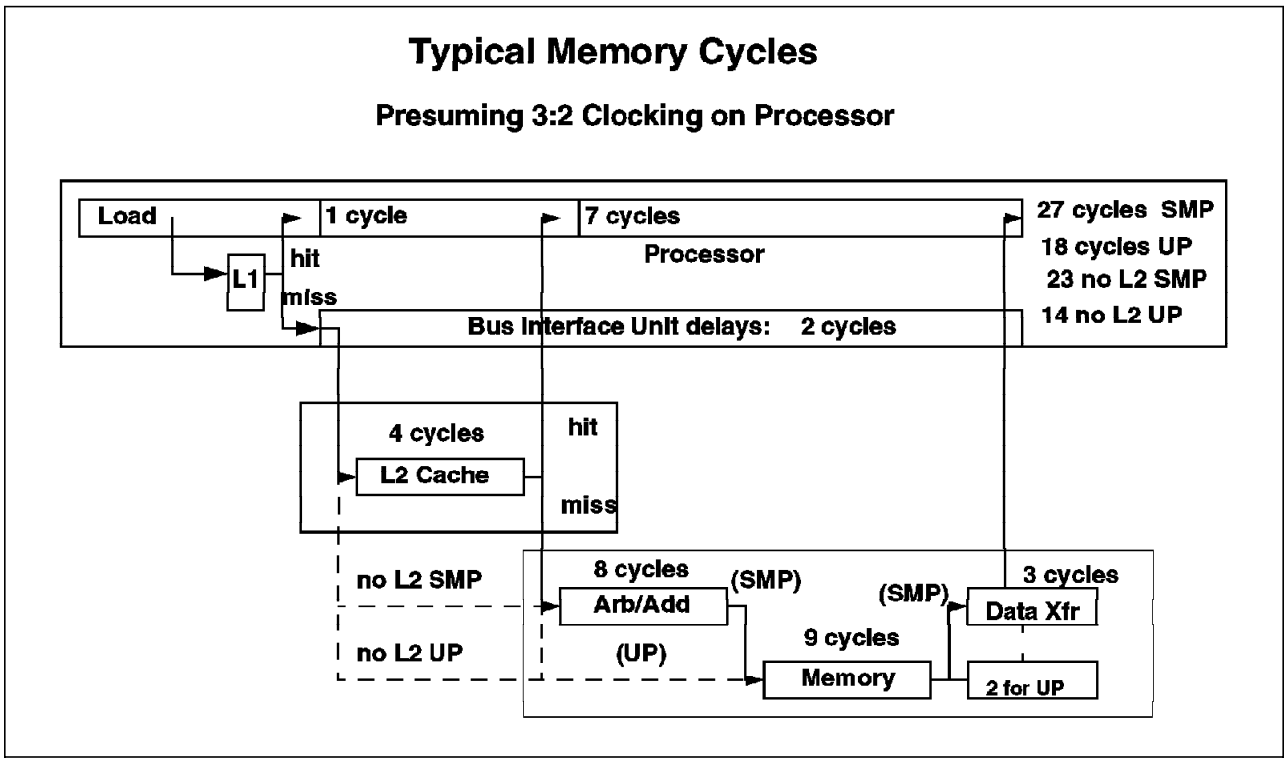


Figure 21. Typical Memory Cycles

In Figure 21, you can see that when there is a hit in L1, it takes only one cycle to access the data.

- If the system does not have any L2 cache, it takes 14 cycles on a uniprocessor to load data from the memory to the processor and 23 cycles for an SMP.
- If the system has an L2 cache, it takes 7 cycles to access data if it is already in the L2 cache (cache hit in L2), but it takes 18 cycles for a UP and 27 cycles for an SMP to access data if there is also a cache miss in L2. Note that there is a two cycle delay between the processor and L2 or the memory.

Figure 21 shows the memory cycles required for getting data from the memory subsystem on a typical system. A 3:2 clocking rate on the processor means that when the processor runs at 100 MHz, the system bus runs at 66 MHz. The 3:2 is the frequency ratio between the processor frequency and the system bus frequency. Phase locked loop (PLL) technology is used to match the bus and processor operating frequencies.

4.2.5 Miss Rate Penalty

Since one of the main differences between a commercial environment and a scientific environment is the miss rate, it is important to understand the effect of a high miss rate for the performance of a system.

Let us take a 100 MHz processor without an L2 cache, and let us also assume that the measured infinite cache CPI is 1.3 on that processor.

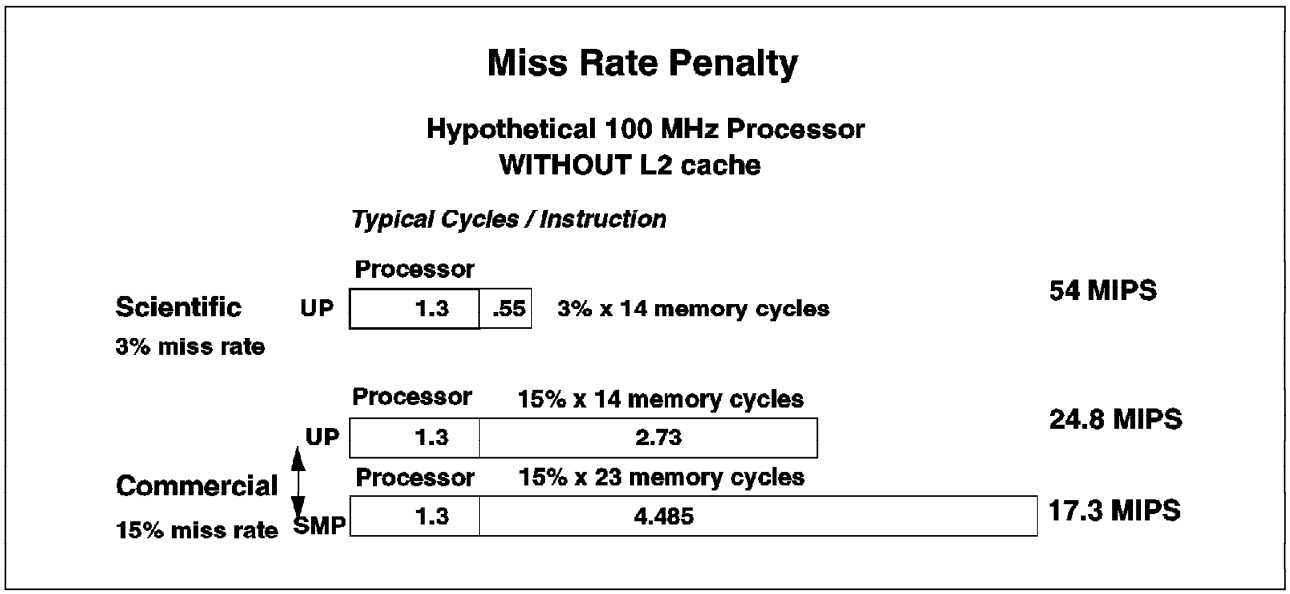


Figure 22. Miss Rate Penalty

In an engineering and scientific environment, the L1 miss rate is around 3 percent. This means that 3 percent of the time, you will need 14 cycles on a UP to access your data from the memory. In this case, the total number of CPI will be:

```
1.3 + (0.03 x 14 x 1.3) = 1.3 + 0.55 = 1.85 CPI.
The "cost" of accessing the real memory is 0.55 CPI.
```

Note: The first 1.3 value is the infinite cache CPI, which can be measured. The second 1.3 value in this calculation is the average number of memory requests per instruction. This second value comes from typical instruction mixes where about 30 percent of instructions are either LOADs or STOREs. Each instruction fetch consumes one memory reference. Adding 0.3 memory references due to LOADs and STOREs results in an average value of 1.3 for the average number of memory references per instruction.

At 100 MHz, the machine will deliver 100/1.85 = 54 MIPS (millions of instructions per second).

In a commercial environment, where the miss rate is usually around 15 percent, it will take 14 cycles for a UP to access data from the memory. For an SMP, you will need 23 cycles to access data from the memory.

Therefore, for a UP, the number of CPI will be:

```
1.3 + (0.15 x 14 x 1.3) = 1.3 + 2.73 = 4.03 CPI.
This corresponds to 100/4.03 = 24.8 MIPS.
```

At 100 MHz, the UP system will deliver only 24.8 MIPS. The higher miss rate lowers the performance of the system by 54 percent (24.8 MIPS vs. 54 MIPS).

For an SMP, the number of CPI will be:

```
1.3 + (0.15 x 23 x 1.3) = 1.3 + 4.485 = 5.785 CPI.
This corresponds to 100/5.78= 17.3 MIPS.
```

At 100 MHz, the SMP system will deliver only 17.3 MIPS per processor. Therefore, a high miss rate lowers the performance on both UP and SMP systems. SMPs have a disadvantage due to the higher number of cycles required to access data from memory. Figure 22 on page 53 shows the miss rate penalty for UP and SMP systems that do not have an L2 cache.

4.2.6 L1 Cache Size

Scientific Environment: Usually, for the scientific environment, an L1 instruction cache (I-cache) size of 32 KB is the optimal value. Having a bigger size will increase the hardware complexity in return for a small performance gain.

For the L1 data cache (D-cache), the optimum value is 32 KB for most of the scientific applications. However, with 32 KB L1 D-cache, some scientific applications like Les (chemistry application) still have high miss rates (about 15 percent). This kind of application requires a larger L1 D-cache to achieve a lower miss rate and good performance. In this case, typical L1 D-cache should be about 128 KB to 256 KB. The P2SC processor with 128 KB D-cache is well-suited to running such applications.

Scientific applications are more likely to take advantage of instructions brought into the cache. The L1 I-cache miss rate is determined by the typically small number of instructions that are required to do the work in comparison to the size of the cache.

In the scientific environment, miss ratios are generally more important for D-cache than for I-cache because scientific programs have a small footprint in memory.

Commercial Environment: For the commercial environment, having an L1 I-cache size of 32 KB is an optimal value. Commercial workloads usually have much higher instruction cache miss rates overall than scientific workloads.

The L1 D-cache miss rates are generally lower than the I-cache miss rates in commercial environments. An increase in the D-cache size does not yield a big performance improvement.

Multiuser commercial workloads experience higher instruction cache miss rates because of larger instruction footprints and process switch activity. For small L1 I-caches (less than 32 KB), the size of the instruction footprint dominates cache performance. As the cache size increases, the effect of multiple processes and process switch intervals becomes a factor. For example, a commercial application process is typically only in the running state for a relatively short period of time before another process switches in and takes over the cache. By the time the first process is reactivated, many of its instructions have been

replaced. Larger cache sizes increase the probability that the process will find the cache in a useful state when it returns.

Commercial workloads experience higher miss rates in the I-cache versus D-cache, whereas the opposite is true for scientific workloads.

4.2.6.1 Cache Line Size and Associativity

Experiments have shown that longer lines and larger caches sizes can help the cache performance of multiuser commercial workloads more than increasing the associativity. The benefit for single-user scientific workloads is dependent on code and data structures.

4.2.7 Effect of L2 Cache

Adding an L2 cache to a UP or to an SMP system can lower the time spent accessing memory data. This has a different effect depending on commercial or scientific environments. Typically, in case of an L1 cache miss, the probability of finding the data in L2 is 80 percent. This means the miss rate is 20 percent in L2 cache.

Scientific Environment: On a UP system equipped with an L2 cache, it takes seven cycles to get data from L2 and 18 cycles to get data from the memory. We can calculate the average number of additional cycles per instruction needed in case of an L1 cache miss as follows:

```
0.03 x 0.8 x 7 x 1.3 + 0.03 x 0.2 x 18 x 1.3 = 0.22 + 0.14 = 0.36 CPI
versus 0.55(accessing the memory without L2 cache).
The total CPI number is 1.3 + 0.36 = 1.66 CPI
```

Adding L2 cache to a UP system saves only 0.19 CPI (0.55 CPI - 0.36 CPI) in a scientific environment. An L2 cache does not significantly improve the performance of the system in a scientific environment (100 / 1.66 = 60.2 MIPS instead of 54.0 MIPS). For example, the 3BT (POWER2) without L2 cache is rated at 3.14 SPECint95. With L2 cache of 1MB, it is rated at 3.21 SPECint95.

Commercial Environment: On an SMP equipped with an L2 cache, it takes seven cycles to access data from L2 and 27 cycles from the memory. Thus, with a 15 percent L1 miss rate and a 20 percent L2 miss rate, the number of additional cycles per instruction is:

```
0.15 x 0.8 x 7 x 1.3 + 0.15 x 0.2 x 27 x 1.3 = 1.09 + 1.05 = 2.14 CPI
versus 4.485 (accessing the memory without L2 cache).
The total CPI number is 1.3 + 2.14 = 3.44.
This corresponds to 100/3.44 = 29 MIPS.
```

In this case, the L2 cache has a great effect and can increase the performance of the system up to 67 percent (17.3 MIPS vs. 29 MIPS). Figure 23 on page 56 shows the L2 cache effect for a UP in a scientific environment and the L2 cache effect of an SMP in a commercial environment.

Conclusion: As the L1 miss rate increases, adding an L2 cache can greatly improve the performance of the system.

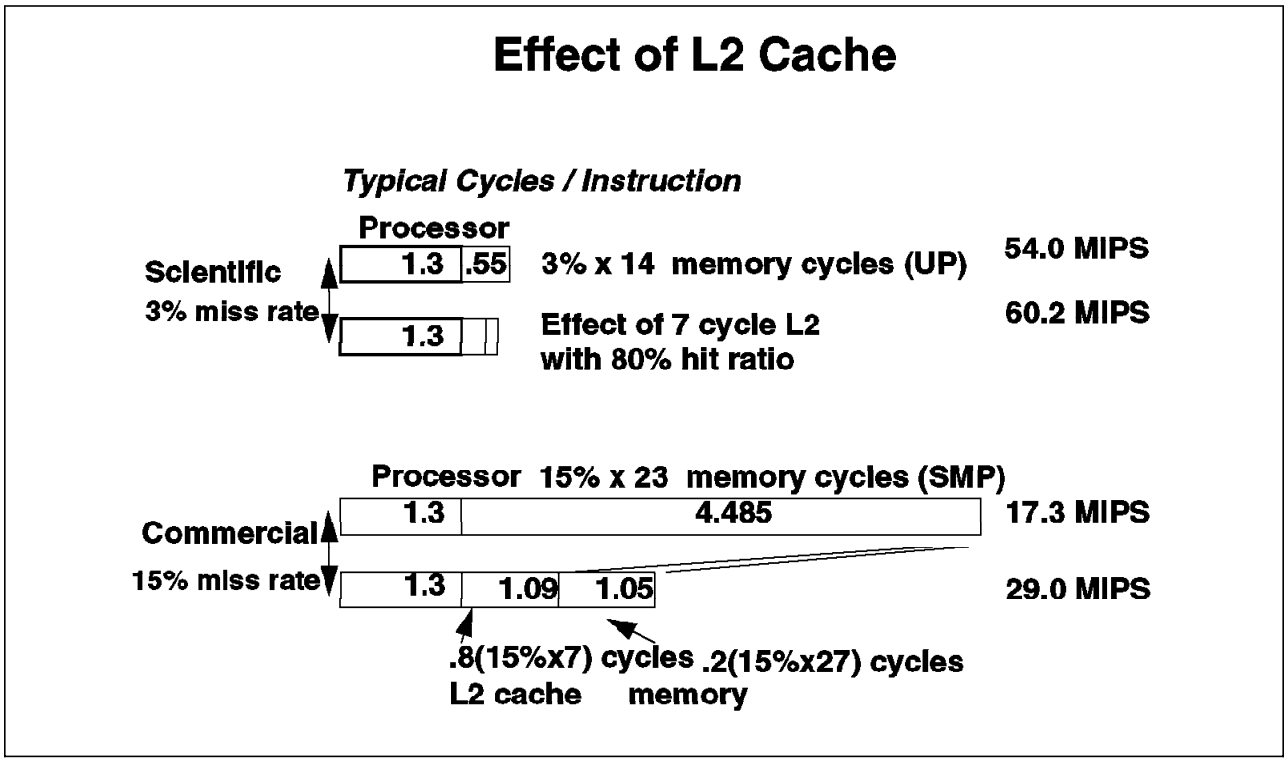


Figure 23. Effect of L2 Cache

4.2.8 L2 Latency vs. Processor Speed

Adding an L2 cache to a UP or SMP that is working at the same speed as the processor can improve performance. In this case, the latency is four cycles instead of seven cycles. Adding this kind of L2 cache has different effects for scientific and commercial environments. Typically, for an L1 cache miss, the probability of finding the data in L2 is 80 percent. This means the miss rate is 20 percent in L2.

Scientific Environment: On a UP system equipped with an L2 cache that is working at the same speed as the processor, it takes four cycles to get data from L2 and 18 cycles to get data from memory. The average number of additional cycles per instruction needed in case of an L1 cache miss is calculated as follows:

```
0.03 x 0.8 x 4 x 1.3 + 0.03 x 0.2 x 18 x 1.3 = 0.12 + 0.14 = 0.26 CPI
versus 0.36 (accessing the memory with a normal L2 cache).
The total CPI number is 1.3 + 0.26 = 1.56 CPI.
```

Having an L2 cache working at the same speed as the processor on a UP system saves only 0.1 CPI (0.36 CPI - 0.26 CPI) in a scientific environment. An L2 cache running at the same speed as the processor does not significantly improve the performance of the system in a scientific environment (100 / 1.56 = 64.1 MIPS instead of 60.2 MIPS for a normal L2 cache).

Commercial Environment: On an SMP equipped with an L2 cache working at the same speed as the processor, it takes four cycles to access data from L2 and 27 cycles from memory. Thus, with a 15 percent L1 miss rate and a 20 percent L2 miss rate, the number of additional cycles per instruction is:

```
0.15 x 0.8 x 4 x 1.3 + 0.15 x 0.2 x 27 x 1.3 = 0.62 + 1.05 = 1.67 CPI
versus 2.14 (accessing a normal L2 cache). The total CPI number
is 1.3 + 1.67 = 2.97. This corresponds to 100/2.97 = 33 MIPS.
```

In this case, having an L2 cache working at the same speed as the processor can increase the performance of the system up to 16 percent (33 MIPS vs. 29 MIPS).

Conclusion: As the L1 cache miss rate increases, having an L2 cache working at the processor speed increases performance.

4.2.9 Effect of Processor Speed

If the processor speed is doubled, the effect of L1 cache on CPI will be the same as calculated above. (Note that this is true if L1 is embedded in the processor chip itself.) L2 cache must run at a speed close to the processor speed to maintain good performance. This can be achieved because the technology used to build such L2 caches comes from the technology used to make processor memory. Of course, using such an L2 cache will increase the overall price of the system. In this case, typical L2 latency time should be between two and five cycles, depending on the processor bus frequency. For very high processor frequency, a 1:1 ratio between processor and L2 cache is hard to achieve. The typical ratio is 3:2 or 5:4.

Let us now assume that for a bus running at 66 MHz, it takes eight cycles to access memory. For a processor running at 100 MHz and working with a bus running at 66 MHz, the ratio will be 3:2 (1.5 * 66 = 100 MHz). Thus, accessing the memory will require 1.5 * 8 = 12 CPU cycles.

Scientific Environment: For a processor running at about 200 MHz, this will give a 3:1 ratio with the memory bus (a bus running at 66 MHz). In that case, the main memory latency will be about 24 (3 * 8) processor cycles. Accessing the L2 cache should still require about 5 processor cycles. In that case, for scientific environments, we will have:

```
0.03 x 0.8 x 5 x 1.3 + 0.03 x 0.2 x 24 x 1.3 = 0.156 + 0.187 = 0.343 CPI

The total CPI number is 1.3 + 0.343 = 1.643
This corresponds to 200/1.643 = 122 MIPS.
```

In Table 5 on page 58, based on the previous formula, we summarize performance gains for increasing processor frequency.

Processor Speed	Memory Latency	MIPS	Delta Improvement	Ratio to 100 MHz
100 MHz	12 cycles	65 MIPS	N/A	N/A
200 MHz	24 cycles	122 MIPS	88%	1.9
300 MHz	36 cycles	173 MIPS	42%	2.7
400 MHz	48 cycles	219 MIPS	27%	3.4
500 MHz	60 cycles	260 MIPS	19%	4

Table 5. Processor Speed Effects for the Scientific Environment

For scientific environments using high-speed processors, even if the cache miss ratio is low, the high latency time of the main memory will greatly reduce the overall performance of the machine. The speed of the bus (66 MHz) will also reduce the performance of processors with high frequencies.

Commercial Environment: For commercial environments with a processor running at 200 MHz, we will have:

```
0.15 x 0.8 x 5 x 1.3 + 0.15 x 0.2 x 24 x 1.3 = 0.78 + 0.936 = 1.716 CPI

The total CPI number is 1.3 + 1.716 = 3.016.
This corresponds to 200/3.016 = 66 MIPS.
```

In Table 6, based on the previous formula, we summarize performance gains for increasing processor frequency.

Processor Speed	Memory Latency	MIPS	Delta Improvement	Ratio to 100 MHz
100 MHz	12 cycles	39 MIPS	N/A	N/A
200 MHz	24 cycles	66 MIPS	69%	1.7
300 MHz	36 cycles	86 MIPS	30%	2.2
400 MHz	48 cycles	101 MIPS	17%	2.59
500 MHz	60 cycles	113 MIPS	12%	2.9

Table 6. Processor Speed Effects for the Commercial Environment

For commercial environments using high-speed processors and with a high cache miss rate, the high latency time of the main memory will greatly reduce the overall performance of the machine. The speed of the bus (66 MHz) will also reduce the performance with high processor frequency.

Conclusion: This shows that the higher the miss rate, the smaller the benefit from increasing the processor speed. When reaching high processor frequency, continuing to increase the processor speed will lead to little performance gain.

4.2.10 Memory/Cache Effects

The memory design is also an important factor for performance, especially for an application that is memory intensive (has a high cache miss rate).

Suppose we do the following computation:

```
DO i = 1,n
y(i)=y(i)+a*x(i)
END DO
```

and each access to a new cache line of either array “x” or “y” results in a cache miss and, therefore, a memory access.

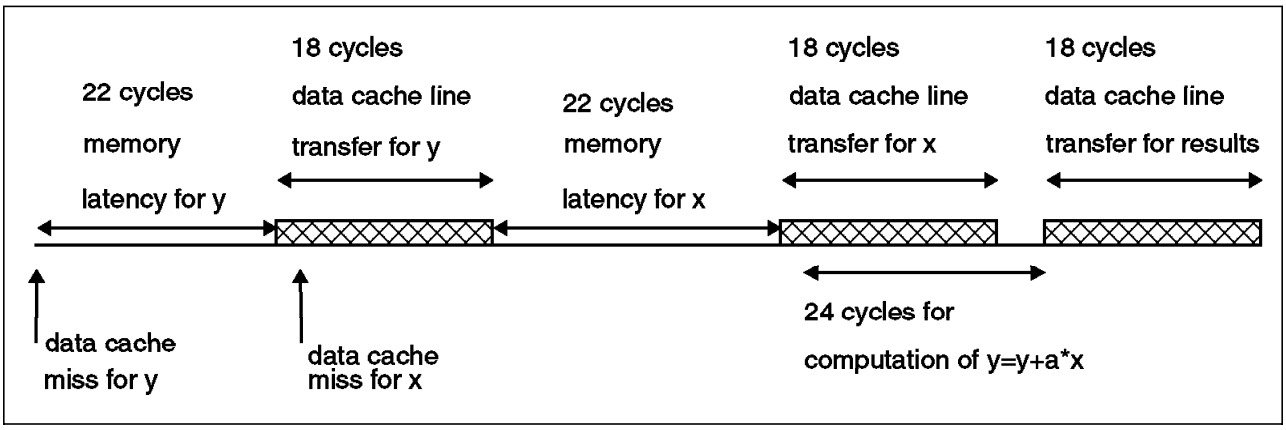


Figure 24. Memory Timeline Access

Figure 24 reflects the following sequence:

- The processor needs a “y” operand. A cache miss occurs.
- Assume it takes 22 CPU cycles to bring the first “y” operand of the missed cache line into the CPU cache.
- Assume it takes 18 CPU cycles to transfer the remaining data elements in the 256 B data cache line containing the “y” operand (8 * 2 cycles per 32 B + 2 dead cycles) into the cache line.
- The processor needs an “x” operand. A cache miss occurs.
- Assume it takes 22 CPU cycles to bring the first “x” operand of the missed cache line into the CPU cache.
- Assume it takes 18 CPU cycles to transfer the remaining data elements in the 256 B data cache line containing the “x” operand (8 * 2 cycles for 32 B + 2 dead cycles) into the cache line.
- As soon as the first operand arrives in the data cache, the computation can start (even if the data cache line transfer is not complete)
- It takes 18 CPU cycles to store the cache line to memory.
- The computational time is hidden by the data cache line transfer, and we assume 0 in this example.

In that case, the total number of CPU cycles is: 2 * (memory latency) + 3 * (data cache line transfer) + (computations not hidden under two data cache line transfers).

This gives 2 * 22 + 3 * 18 = 98 cycles to compute 32 values.

Note that the computation takes only 24 cycles; the bulk of the time is spent in the memory subsystem.

Consider now the same example but with a 64 B line size. In that case, we will have to fetch four times more data from memory. For each data line transfer, the latency time is: 2 * (2 CPU cycles for 32 B) + 2 dead CPU cycles = 6 CPU cycles.

We will have: 2 * 4 * (memory latency) + 3 * 4 * (data cache line transfer).

This gives 8 * 22 + 3 * 4 * 6 = 248 cycles to compute eight values.

If the cache line size is cut from 256 B to 64 B, the number of cycles required to compute the same number of operands goes from 98 cycles to 248.

Also note, if the width of the bus is cut from 32 B (256 bits) to 8 B (64 bits), then the number of cycles increases from 98 to 194. (In that case, the transfer for a data cache line is 50 cycles.)

This shows the dramatic effect of memory/cache system parameters on the performance of memory-intensive applications.

4.3 Storage

The evolution of the RS/6000 system architecture has included many changes in power management, processors and memory. Also, the I/O requirements have increased, and faster response times are required. In order to reach the response times and the throughput levels required today, some changes have been made to adapter and disk technologies.

The traditional SCSI and SCSI-2 interfaces, used in all the old RS/6000 system models, have begun to reach their limits. The disks themselves have realized large-scale performance improvements, placing greater demands on the disk interface bus. Disk seek times have been reduced to less than 10 milliseconds (ms). The rotational speed has doubled (3600 rpm for old SCSI disks to 7200 rpm now), and the disk data transfer rate has increased from less than 5 MB/s to over 15 MB/s.

Better response time is not the only need in an RS/6000 system. The demand for large storage solutions is growing faster than before. Applications today demand larger data objects, so complex databases and historical information issues imply the need for total disk space in excess of 10 GB. Historical information maintenance, client/server computing and mission-critical applications also demand higher levels of availability, fault tolerance, higher levels of performance and greater connectivity options.

4.3.1 Performance View

Fast processors running applications that cause large numbers of I/O disk drive accesses can become I/O bound and degrade system throughput. To sustain performance, computer systems have been using ever-larger memories. This could be fine, but memory is one of the most expensive components in the system, and its use must be rational. To use large amounts of memory for more file system buffering works well for systems that have locality of reference. But applications dominated by a high rate of random requests for short records or by a smaller number of requests for massive records must still face the underlying disk performance problem.

Amdahl's Law can be used to explain how the performance of an application can be effected by increasing only the processor speed or both processor and disk speeds:

$$EAS = 1 / ((1 - FFM) + FFM / SFM)$$

Figure 25. Amdahl's Law

Where:

EAS Effective application speed-up

FFM Fraction of the work in the faster mode (using the fastest components such as processor and memory)

SFM Speed-up of the faster mode

The law predicts that, if we have an application running 80 percent of its time on the processor (FFM=0.8), and 20 percent in I/O operations, then increasing the processor speed by a factor of 2 (SFM=2) results in an increment of 1.67 in the application speed (EAS = [1/**(**(1-0.8)+(0.8/2)**)**]). When the processor is 16 times faster, the same application will be only 4 times faster.

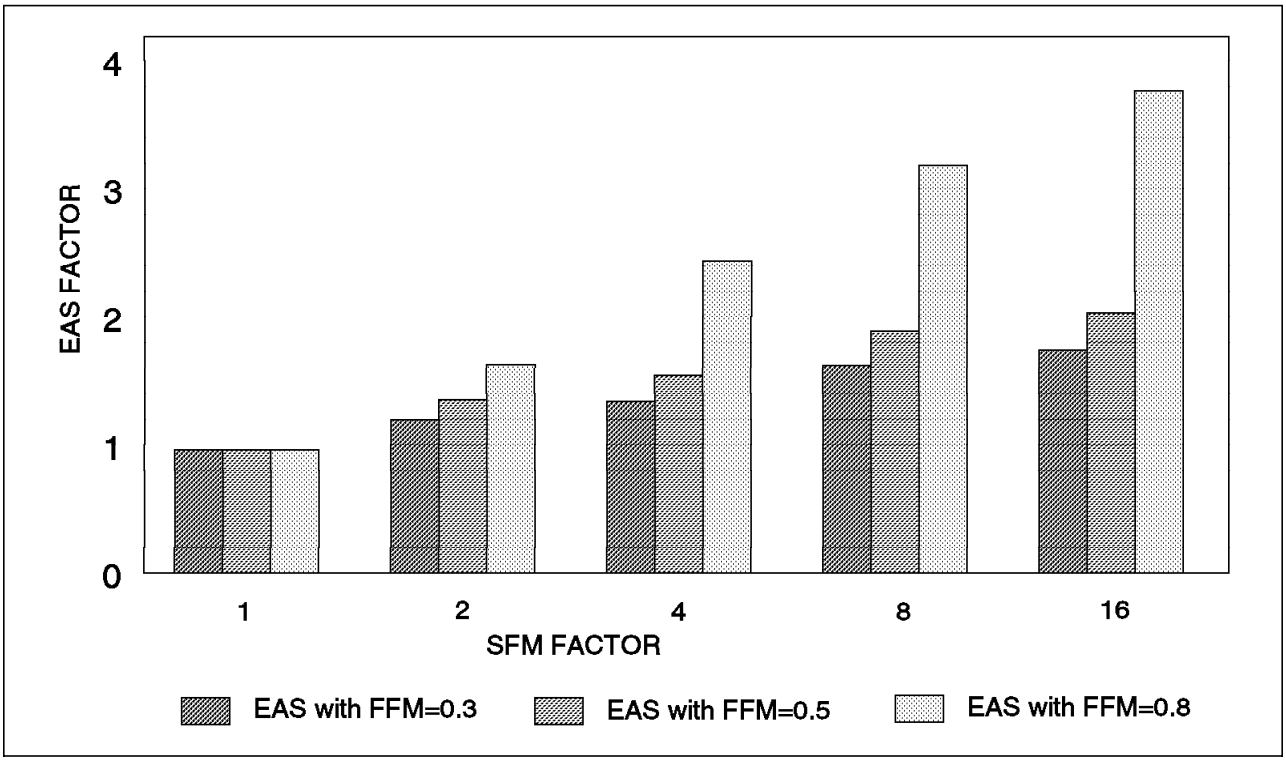


Figure 26. Speed-Up Comparison

Note that for applications with high I/O, the most effective speed-up tends to be 1/(1-FFM). For this example, we theoretically could increase the processor speed by a factor of 1000, but the acceleration of the application will be only about 5.

Levels of Storage: When we consider storage as the slower mode that an application can work in, several components of the system architecture are involved: I/O subsystem, bus controller (MCA, PCI, ISA), disk subsystem adapters (SCSI, SSA, Serial Link), disk array controllers, disk drive electronics and, finally, disk drive mechanics. The lower the level of the component, the slower speed it has.

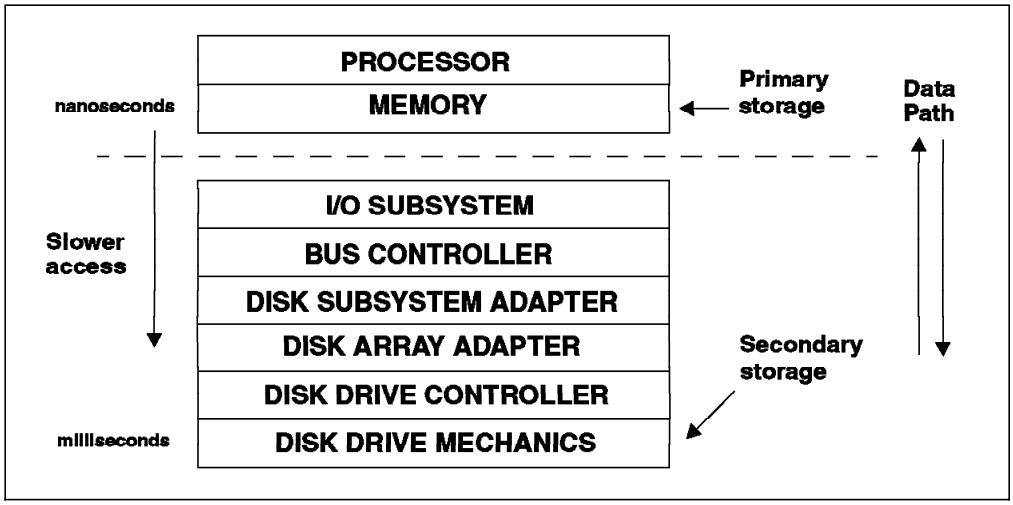


Figure 27. Levels of Storage

How an I/O Request is Processed: When a process needs to reach data from a data file, it uses a system call to the operating system. The operating system puts the request of the process for I/O in a queue managed by a specific driver for the device. All requests are organized in the queue, using a seek optimization algorithm, so the disk heads can easily find all the data requested. The device driver translates the request for data into commands for the disk drive. These commands will manage the disk mechanics to find the data across the disk surface.

Once the request is being processed by the disk drive and the head is moved to the correct track, the data is transferred to the drive controller. The device driver takes the data from the controller when a hardware interrupt notifies the operating system of the result of the operation. The data is finally transferred to process memory (via DMA), and the process is awakened and placed on the run queue to wait for the processor.

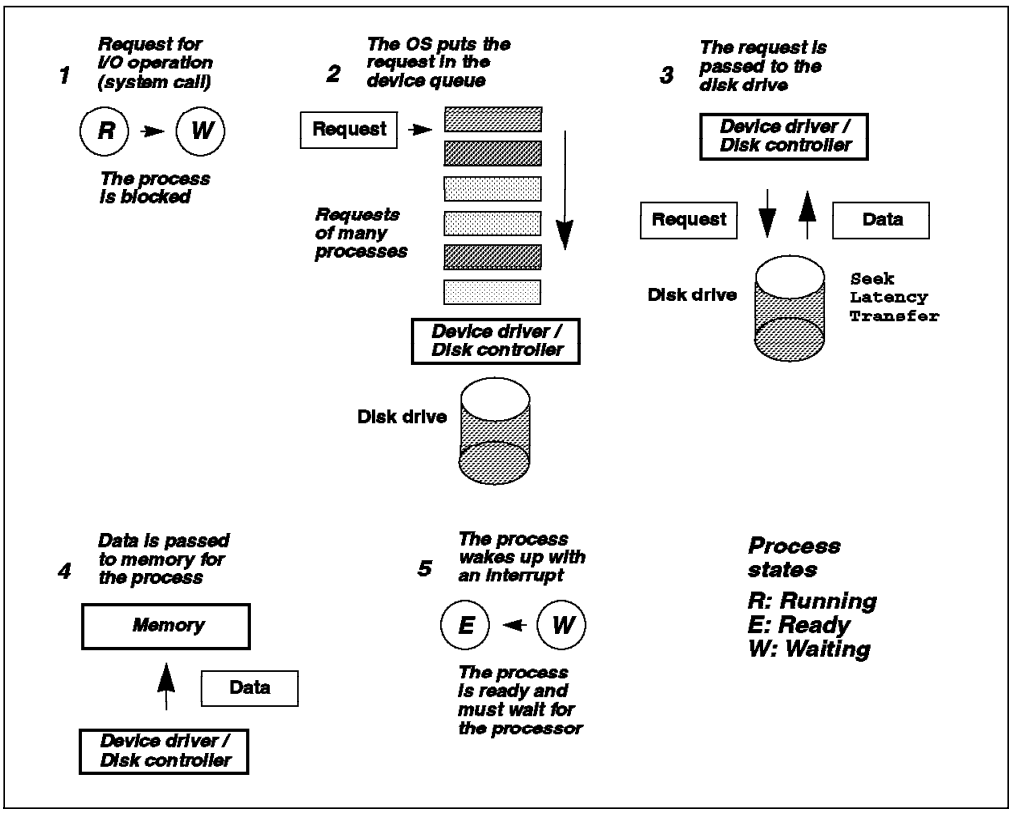


Figure 28. The I/O Request Path

The workload for a single process is not meaningful when analyzing an entire system except when the process is the only active one. Secondary storage of a system is accessed many times by different applications, and some requests can be concurrent. The read or write commands to be processed by a disk drive could be either a few or many thousand bytes each. There is no specific pattern for the order in the queue of the device.

Random access When the workload is composed of many accesses to many files simultaneously in a system and most of the accesses are only a few bytes. If the data is spread over the disk in a random pattern, then most of the accesses would be over different tracks of the disk. This implies that the disk is spending most of its time in setting up (seeking) for the next transfer. This is a typical random-access workload.

Sequential access A workload is considered to have sequential access if most of the requests to the disk drive are for large and continuous byte streams for a few files and if the data is allocated sequentially in continuous portions of the disk. Most of the reads or writes would be over continuous tracks, and the disk would not spend much time setting up for the next transfer.

Since the I/O requests of the processes are translated in system calls, many I/O accesses will demand significant processor time. The more commands executed per time interval, the greater the processor utilization. It must be noted that intensive I/O does affect computation-intensive applications when both I/O and application processes are running in parallel. For a heavily loaded system

performing 100 to 200 physical I/Os per second, this might result in 15 to 25 percent of the processor time needed for servicing the I/O requests alone.

In many environments, the overall performance of an application is bound by the speed at which data can be accessed from secondary storage. A good predictor of the overall performance of a fixed disk is the rate at which it can read or write these disk blocks.

For most applications following a random-access pattern, the access time of a disk is the best way to measure its performance, specially when the access is for little data blocks.

For applications following a sequential access pattern, a good indicator of disk performance is the throughput of the disk adapter at peak rates.

The degree to which a disk may be utilized depends on the way it is used. If the disk drive is dedicated to a single process, even a 100 percent device busy state is generally no reason for concern. In fact, it might be desirable. This situation is quite different in the case where multiple users are working with the disk at the same time. Here, a 40 percent device busy state should be investigated.

How a Disk Works: Disk drives only understand special commands given by their controllers. So a SCSI disk drive only works with SCSI commands and an SSA disk with SSA commands. The manner in which the hardware of the disk drives works is the same for all of them. Each request for reads or writes has a virtual address of the data on the disk (sector address), an action (read or write), the size of the string to be processed, and the string to be written (if any).

All the actions for reading or writing bytes to the disk drive are coordinated by the kernel and are transparent to the processes. They only attempt to interpret the result and data of the *read()*, *write()*, *seek()*, *open()* or other functions when the kernel returns them.

System calls for I/O generate four main tasks:

1. To evaluate where the bytes are or will be located on the disks (disk drive, platter, head, track, sector).
2. To create a command for the disk drive hardware.
3. To activate the disk arm to position the correct head over the track to be read or written.
4. To read/write and transfer the bytes to/from memory.

The last two tasks are handled by the disk drive (except the transfer part).

The access time of a disk consists of three components:

Seek time A seek is the physical movement of the head at the end of the disk arm from one track to another one. The time for a seek is the necessary time for the disk arm to accelerate, to travel over the tracks to be skipped, to decelerate and finally to settle down and wait for the vibrations to stop while hovering over the target track. The total time the seeks take is variable. The average seek time is used to measure the general disk capabilities, and it is generally lower than 15 ms. The time is 8.5 ms for most of the SCSI-2 Fast/Wide (F/W) disk drives used today.

Latency The rotational latency is the time that the disk arm has to wait while the disk is rotating underneath until the target sector approaches. Rotational latency is, for all practical purposes except sequential reading, a random function with values uniformly between zero and the time required for a full revolution of the disk (less than 10 ms). The average rotational latency is taken as the time of a half revolution, and it is generally lower than 5 ms. For most of the SCSI-2 F/W disk drives used today, this time is 4.1 ms.

Transfer The data transfer time is determined by the time it takes for the requested data block to move through the read/write arm. It is linear with respect to the block size. For a 4 KB page transfer, this time is typically near 1 ms. Most of the SCSI-2 F/W disk drives used today have a media transfer rate between 4.9 and 8.2 MB/s.

The average disk access time is the sum of the averages for seek time and rotational latency plus the data transfer time (normally given for a 512 bytes block). It lies between 12 ms for SCSI-2 F/W disk drives and 26 ms for old SCSI disk drives. The average disk access time generally overestimates the time necessary to access a disk. For random access, seeks tend to be shorter than the average. Typical disk access time is 70 percent of the average.

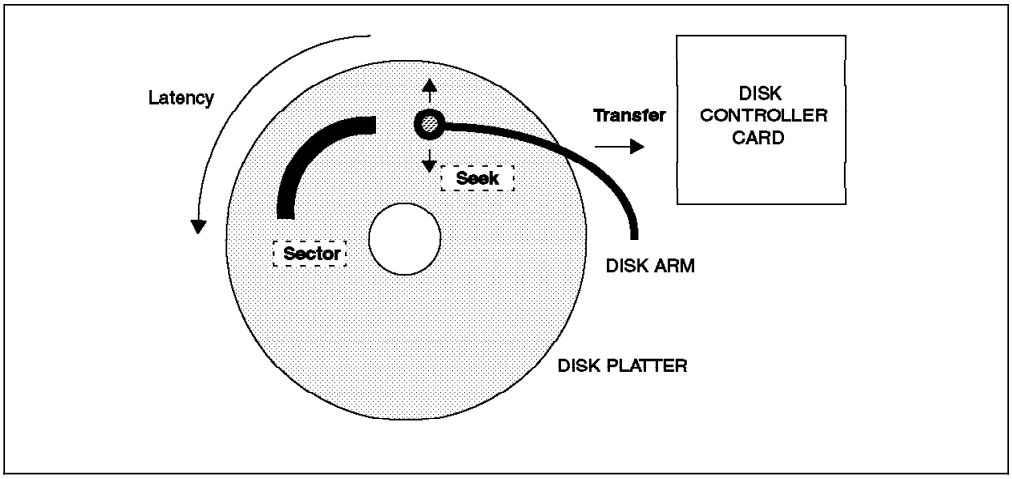


Figure 29. Disk Times

4.3.2 SCSI Technology

SCSI stands for small computer system interface and refers to the original standard approved in 1986. SCSI became one of the first parts of open systems hardware.

SCSI involves parallel transmission of data across a parallel set of wires. These wires carry data and clock signals, and the devices are attached to the set of wires (SCSI cable) forming a bus. The number of wires in a SCSI cable is 50 for SCSI and SCSI-2 and 68 for SCSI-2 F/W. The first SCSI, also named 8-bit SCSI, is limited to handling eight total addresses and can transmit data at 4.5 MB/s. SCSI-2 improves the clock speed over SCSI and can transmit data over the bus up to 10 MB/s. SCSI-2 F/W can handle up to 16 total addresses in a 16-bit bus and can transmit up to 20 MB/s.

Every SCSI device (disk, CD-ROM, tape) is connected to the bus using the same wires as the other SCSI devices. Therefore, data can travel in only one direction

at a time. A SCSI device can process multiple commands simultaneously. An arbitration level is required in the controller to prevent multiple devices from using the bus at a time. As the SCSI adapter needs one address to access the bus, the maximum number of the devices it can support is seven for SCSI and SCSI-2, 15 for SCSI-2 F/W in a single bus or 30 in two buses (AIX V3.2.5 allows up to 14 in two buses).

The SCSI definition uses offline seeks. This means several disks on the SCSI bus can be in their seek or rotational latency phases concurrently without blocking the SCSI controller. Although the SCSI drives can seek offline, they must transfer data by gaining control over the SCSI bus. Thus, only one SCSI device can be transferring data at a time. Thus, the real throughput is limited by the disk drive throughput and by the adapter throughput.

Ultra SCSI is a clock doubled version of the current 8-bit SCSI-2 and 16-bit SCSI-2 F/W standards. The 8-bit Ultra, sometimes called Fast 20, has a maximum data transfer rate of 20 MB/s, similar to the speed of the Fast and Wide interface. The 16-bit Wide Ultra has a top speed of 40 MB/s.

Ultra SCSI is the parallel implementation of the next generation called the SCSI-3 interface family. While those in the SCSI-2 lineup are all parallel, SCSI-3 will support serial standards, including SSA. A current definition of SCSI-3 in parallel mode is Ultra SCSI. SCSI-3 will be defined by the American National Standards Institute (ANSI) standards committee.

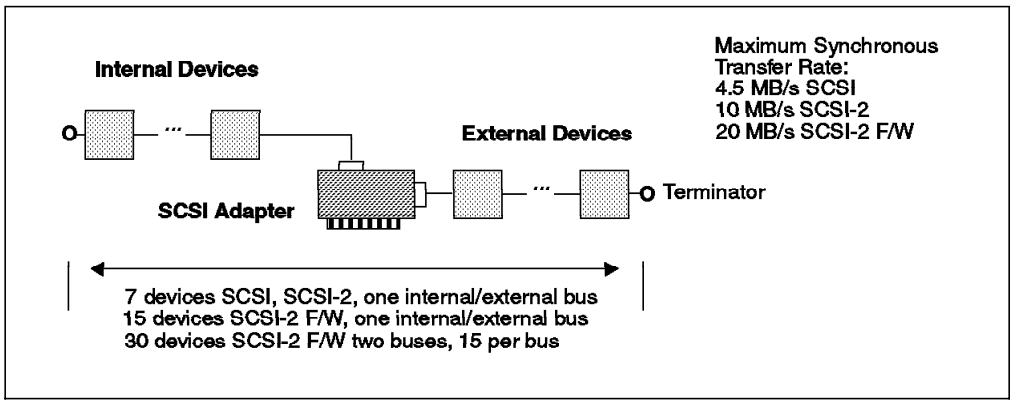


Figure 30. SCSI Adapter

There are two different ways to use SCSI adapters and devices: Single Ended (SE) and Differential.

SCSI Single Ended This is the traditional bus, in which only single-ended devices can be attached. Only one SCSI adapter can control the bus.

SCSI Differential In a differential bus, two or more adapters can coexist. Because of the possibility of using internal devices and the existence of only one bus, SCSI differential adapters must be selected to use either the internal and external chains or two external ones.

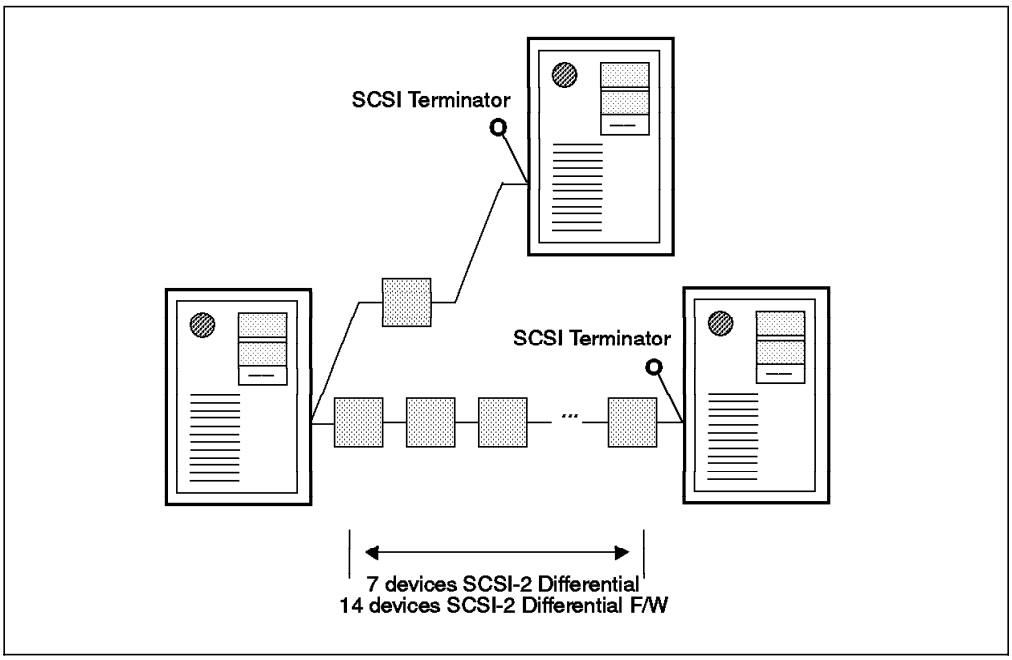


Figure 31. SCSI-2 Differential Sample

There are some considerations for SCSI configurations:

- Differential and SE devices and adapters cannot be mixed because their bus interfaces are electrically incompatible.
- SCSI bus performance is subject to degradation when the number of connected devices increases.
- The shared, arbitrated bus architecture limits the number of devices and distances.

4.3.3 Serial Link Storage

There are two ways to send data from an adapter to a device: serial or parallel. In a parallel implementation, there are wires to transmit and receive data and wires for clock signals. Those clock signals are used to synchronize the transmission of bits in all the wires of the cable. Multiple bits are on the cable at the same time. In a serial implementation, data is sent bit by bit, in a string over the wire. The clock signal may be self contained in the data in a serial transmission.

IBM has developed a special serial data transmission technology, used initially for storage devices. The first implementation was the well-known serial link for the 9333 units used in the RS/6000 systems.

The 9333 disk subsystems are attached to the serial link high-performance adapter with a four-wire cable, point to point. Each connection can handle up to four disk drives, and an adapter can handle up to four independent subsystems. Up to eight disk drives can be handled with no performance degradation.

A serial link disk subsystem can also be connected to up to eight different adapters, to allow sharing of storage between systems and the configuration of high-availability solutions.

4.3.4 Serial Storage Architecture (SSA)

The IBM serial storage architecture (SSA) expands the serial link capabilities and is accepted as a standard by the ANSI committee. The standard uses a similar command set as SCSI, and the communication protocol is designed to allow applications using SCSI definitions to work with the SSA bus.

SSA has the basics of serial link but is enhanced to support fault tolerance and high-performance loops with a large number of devices. An SSA adapter has four full-duplex ports. Two are necessary to configure a loop; the other two can be used for another loop either with other devices or as a backup of the first one.

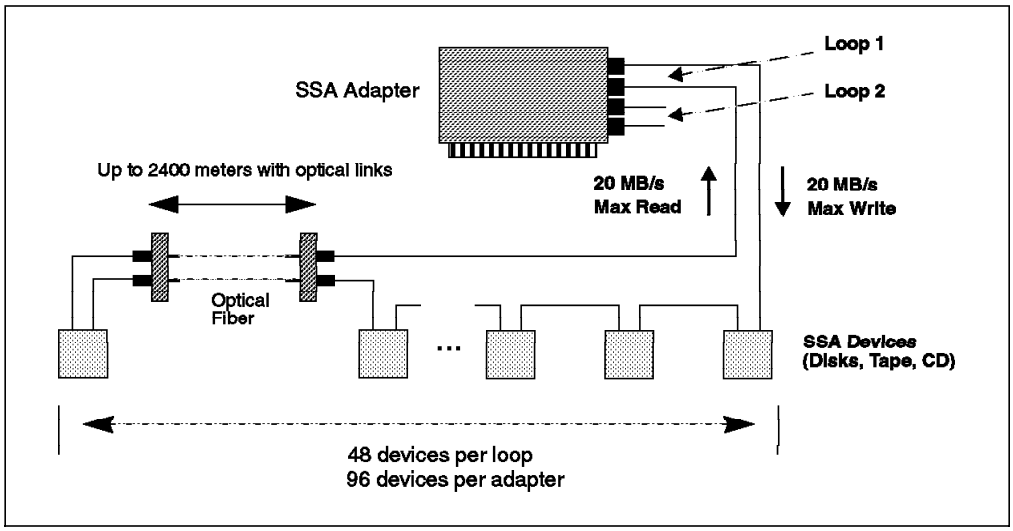


Figure 32. SSA Adapter for the RS/6000

SSA has the following characteristics:

- Support for up to 127 devices (48 or 96, depending on the SSA adapter, for RS/6000 implementations) and up to 20 meters of cable between devices using four-wire SSA cables or up to 2400 meters using fiber-optic extenders.
- The maximum bandwidth is 40 MB/s per adapter full-duplex port (considering 50 percent reads and 50 percent writes). Therefore, data can be transferred up to 20 MB/s in the read direction and 20 MB/s in the write direction for a single SSA device. The maximum bandwidth per adapter pair is 80 MB/s (one loop) and 160 MB/s per adapter.
- An alternate path of communication between the adapter and a device can be defined to act in the event a cable or SSA port on a device fails.
- Dual port, full-duplex architecture in the SSA architecture allows devices to be connected in configurations with no single point of failure.
- Multiple devices can transmit data simultaneously by a characteristic called spatial reuse.
- A device in an SSA loop can be added or removed via hot plug without losing access to any other device in the loop.
- All the loops are auto configured so that no special address switches are required in SSA devices.

An SSA fiber-optic extender is available to support longer distances in a loop configuration. This feature supplies a pair of optical converters that transform

electrical SSA signals into optical signals for transmission via customer-supplied optical fibers. This feature allows the maximum distance between SSA nodes to be increased from 20 meters to 2400 meters for use in campus type installations. One fiber-optic set is required for each SSA loop segment.

4.3.5 Fiber Channel Arbitrated Loop (FC-AL)

FC-AL is a new set of interfaces and protocols that intends to support linking storage devices using high-speed network communications hardware. FC-AL has standards supported by a trade association in which IBM, Hewlett-Packard, Sun and others are members.

The fiber channel protocol can be implemented in three different ways at the physical and electrical interface levels: point to point, switched, and an arbitrated loop protocol. Actually, there are some prototypes of adapters and storage devices using the FC-AL protocol.

FC-AL has the following characteristics:

- The bandwidth of FC-AL links is 100 MB/s using fiber channel standards.
- Up to 126 devices can be connected to a single FC-AL loop.
- Devices are auto configured in an FC-AL loop.
- The hot swappable characteristics needed for a fault-tolerant environment are achieved using a port bypass circuit technique.
- Power requirements for FC-AL interface circuitry are high.
- The FC-AL loop-based scheme allows only one device to use the loop at a time, reducing the effective data throughput that can be achieved as more devices are added.

4.3.6 RAID Technology

RAID stands for redundant array of independent disks. The premise is to take many small disk drives and organize them so that data is spread across them. RAID technology involves the use of electronic controller hardware or software, or a combination of both, to connect disk drives to a computer system to protect the data when a physical disk drive fails. The idea was published in a Berkeley white paper in 1987 and described five levels of the architecture. RAID 6 was introduced later to enhance the RAID 5 definition, and there is a RAID 7 definition of third parties.

RAID algorithms may be implemented as part of the operating system's file system software, or as part of a disk device driver (common for RAID 0 and RAID 1). They may be performed by a locally embedded processor on a hardware RAID adapter. Hardware RAID adapters generally provide better performance than software RAID because embedded processors offload the main system processor by performing the complex algorithms, sometimes employing specialized circuitry for data transfer and manipulation.

RAID 0 At this level, there is no data protection. All the disk drives are used to stripe data in parallel sectors across them. RAID 0 offers the advantage of higher transfer speed compared with a single disk drive because of the parallel access. A single drive element failure can result in an unrecoverable data loss.

RAID 1 All data is duplicated in different disk drives. This level requires pairs to store all the copies of the data on separate disk drives. It is often referred to as disk mirroring, dual copy or disk shadowing. The subsystem controller writes data simultaneously to both disk sets and reads from the disk set that offers the fastest read.

RAID 2 At this level, multiple dedicated parity disk drives in a hamming code are used to offer data security and reliability. This allows the possibility of recovering data in case of a disk failure without complete data duplication. This level requires that all disks in a group be accessed, even for small transfers, and that the slowest disk finishes before the transfer is complete.

RAID 3 All parity data is stored in a single disk drive by interleaving the parity information at the byte level, reducing the cost to have multiple dedicated parity disks. Like level 2, all the disk drives are synchronized for any transfer and must wait for the slowest disk drive.

RAID 4 Parity data is striped in sectors or blocks across a single disk drive. Data disk drives are not synchronized for the access, meaning multiple reads to disks can be done independently. This permits faster individual disk reads for small transfers and writes accessing the disk. The parity check disk becomes a throughput bottleneck because each write operation must wait until the entire user data and parity are complete.

RAID 5 RAID 5 defines all of the disk drives for parity and data, allowing independent disk access. This avoids the problem of the parity disk bottleneck present in RAID 4, but writes are slower than RAID 1 and high overhead to track the location of parity addresses is introduced. The overhead to update data and their parity is called Level 5 write penalty.

RAID 6 It is like RAID 5, but with additional parity information written that permits data recovery if two disk drives fail. Extra parity disk drives are required, and write performance is slower than a similar implementation of RAID 5.

RAID 7 In 1991, Storage Computer Corporation (StorComp) released RAID 7. The RAID 7 architecture gives data and parity the same privileges. The level 7 implementation allows each individual drive to access data as fast as possible. This is achieved by three features:

- Independent control and data paths for each I/O device/interface.
- Each device/interface is connected to a high-speed data bus that has a central cache capable of supporting multiple host I/O paths.
- A real time, process-oriented operating system is embedded into the disk drive array architecture. The embedded operating system ″frees″ the drives by allowing each drive head to move independently of the other disk drives. Also, the RAID 7 embedded operating system is enabled to handle a heterogeneous mix of disk drive types and sizes.

The most common RAID implementations are: 0, 1, 3 and 5. Levels 2, 4 and 6 have problems with performance and are functionally not better than the other ones. In most cases, RAID 5 is used instead of RAID 3 because of the bottleneck when using only one disk for parity.

RAID 0 and RAID 1 can be implemented with software support only. RAID 3, 5 and 7 require both hardware and software support (special RAID adapters or RAID array controllers).

4.3.7 Adapters for Storage Devices

Adapters to support secondary storage are essential when considering performance. For small configurations, a single SCSI or SSA adapter can support all the I/O for a workload without reaching saturation levels. For application environments that need several disk drives to satisfy the storage needs, more than one adapter may be necessary. It is possible that one disk by itself cannot saturate the adapter, but for many configurations, to avoid a bottleneck on one adapter, it is necessary to spread the I/O workload over various adapters.

The most common SCSI adapters used today in the RS/6000 family are SCSI-2 F/W. In most cases, SCSI or SCSI-2 devices can be connected to SCSI-2 F/W buses but with a loss of efficiency.

IBM SCSI Adapters

Adapter Description	Feature Code	System Bus Type	Max. Synchr. Rate MB/s	Sust. Synchr. Rate MB/s	Max. Asynch Rate MB/s	Max. Devices	Max. External Length Mts	Number of Buses	HACMP Support	Command Tag Queue
Integrated SCSI SE	N/A	Integrated	4.5	N/A	2	7	6	1 I/E	NO	NO
SCSI SE	2828	MCA	4.5	N/A	2	7	6	1 I/E	NO	NO
SCSI SE	2829	MCA	4.5	N/A	2	7	6	1 I	NO	NO
SCSI SE	2835	MCA	4.5	N/A	2	7	6	1 I/E	NO	NO
Integrated SCSI-2 SE	N/A	Integrated	10	N/A	2.5	7	3-6	1 I/E	NO	NO
SCSI-2 SE	2410	MCA	10	N/A	2.5	7	3-6	1 I/E	NO	YES
SCSI-2 Differential	2420	MCA	10	N/A	2.5	7	19	1 E	YES	YES
Integrated SCSI-2 F/W SE	N/A	Integrated	20	14	3.6	15+15	3-6	2 I/E	NO	NO
SCSI-2 F/W SE	2415	MCA	20	14	3.6	15+15	3-6	2 I/E	NO	YES
Enhanced SCSI-2 Diff. F/W	2412	MCA	20	14	3.6	15+15	25	2 I/E	YES	YES
SCSI-2 Differential F/W	2416	MCA	20	14	3.6	15+15	25	2 I/E	YES	YES
SCSI-2 F/W SE	2408	PCI	20	14	3.6	15	6	1 I/E	NO	YES
SCSI-2 Differential F/W	2409	PCI	20	14	3.6	15	25	1 I/E	YES	YES
SCSI-2 F/W RAID	2493	PCI	20	N/A	N/A	15+15+15	See 7131	3 I/E	NO	N/A

Table 7. IBM SCSI Adapters for the RS/6000

Important

For SCSI-2 F/W adapters, up to 30 devices can be connected in both internal and external buses using AIX V4. AIX V3.2.5 limits this capability to 14 devices (up to 7 internal and up to 7 external).

The SCSI-2 F/W PCI RAID adapter implements RAID levels 0, 1, or 5 and supports up to 45 SCSI-2 F/W physical disk drives on three independent SCSI buses. RAID algorithms and cache management procedures are executed at the

hardware level. The adapter also provides internal configuration routines for the management of the logical drives and underlying physical drives.

IBM Serial and SSA Adapters

Adapter Description	Feature Code	Bus Type	Max. Rate MB/s	Sust. Rate MB/s	Max. Devices	RAID Enabled	Info
Serial Link	6212	MCA	40	N/A	16	NO	Single link per port, 4 ports, Old
SSA 4 port	6214	MCA	40	35	48	NO	LOOP, Old
SSA Enhanced 4 port	6216	MCA	40	35	96	NO	LOOP
SSA 4 port RAID	6217	MCA	40	35	96	YES	LOOP
SSA PCI 4 port RAID	6218	PCI	40	35	96	YES	LOOP, available 04/97

Table 8. IBM SSA Adapters for the RS/6000

Either for SCSI or SSA RAID adapters, RAID algorithms are implemented in the adapter, so disk drive units or disk arrays attached to these adapters don't need to have special RAID characteristics.

4.3.8 Storage Products for the RS/6000

4.3.8.1 Disk Drive Units

The RS/6000 storage options involve the use of single disk units suited to be installed alone or in sets in arrays with or without RAID capabilities. For simplicity of discussion, only the performance values of single disk drives are included in the following table.

IBM SCSI Disk Drive Units

Disk Drive Unit	Type	Media Transfer Rate MB/s	Media Max. Transfer Rate MB/s	Average Read Seek Time ms	Average Latency Time ms
1.0 GB (Old)	SCSI	N/A	5	11	6.9
1.0 GB (Old)	SCSI-2	N/A	10	9.5	5.6
2.0 GB (Old)	SCSI-2	N/A	5.2	9.5	5.6
1.1 GB	SCSI-2 F/W	5.5-7.1	14-16	7.2	4.1
2.2 GB	SCSI-2 F/W	N/A	10.2	8.5	4.1
2.2 GB	SCSI-2 F/W	5.5-7.1	14-16	8.5	4.1
4.5 GB	SCSI-2 F/W	N/A	10.2	8.5	4.1
4.5 GB	SCSI-2 F/W	5.5-7.1	14-16	8.5	4.1
9.1 GB	SCSI-2 F/W	N/A	15.4	8.5	4.1

Table 9. IBM Disk Drive Units for the RS/6000

IBM SSA Disk Drive Units

Disk Drive Unit	Type	Media Transfer Rate MB/s	Media Max. Transfer Rate MB/s	Average Read Seek Time ms	Average Latency Time ms
1.1 GB	SSA	5.5-7.1	14-16	7.2	4.1
2.2 GB	SSA	5.5-7.1	14-16	7.2	4.1
4.5 GB	SSA	5.5-7.1	14-16	7.2	4.1
9.1 GB	SSA	N/A	N/A	7.2	4.1

Table 10. IBM SSA Disk Drive Units for the RS/6000

Important

The 9.1 GB SSA disk drives require AIX 4.1.5 or AIX 4.2 and are not supported in AIX 3.2.5.

4.3.8.2 External Disk Drives and Disk Arrays

There is a wide variety of external storage for the RS/6000. These include SCSI external disk drives and SCSI and SSA disk arrays.

An external disk drive is basically a single disk drive enclosed in a little box with a power supply inside, demanding one SCSI address. There is nothing special to add about external disks.

Disk arrays are sets of single disk drives either managed by a disk array controller or with a direct connection path to the host adapter. The RS/6000 offers a variety of disk arrays to support different needs depending on:

Capacity Depending on the workload, a system can demand low or high external storage capacity. The family of storage products for the RS/6000 includes: low-entry disk arrays ranging from 2 to 20 GB, middle capacity disk arrays ranging from 10 to 40 GB and high-end disk arrays ranging from 20 to 145 GB. Most of the RS/6000 disk array options overlap their capacity. In this case, the selection criteria must be evaluating performance and availability for the target system.

Performance The family of storage products for the RS/6000 includes standard SCSI-2 and SCSI-2 F/W differential or single-ended disk arrays, providing different levels of performance due to the use of direct or RAID controller attachments. High-performance SSA units are offered for the RS/6000 in simple, mirrored or RAID-controlled configurations.

Availability Availability implies that the customer's data will be available from the disk subsystem when it is needed. The RS/6000 offers a variety of availability alternatives:

- Standard availability is achieved using state-of-the-art technology processor, controller and disk drive designs that offer highly reliable systems with a minimum likelihood of components failures. Use of parity, error checking and correction and secure data transmission protocols guard against intermittent or "soft" failures.
- High availability allows the customer to apply redundancy to selected components of the storage subsystem, such as the data itself (using

mirroring or RAID), power supplies, cooling fans, subsystem controllers, SCSI adapters or SSA loops.

- The use of multiple host connections to keep the data available and online when a server fails is achieved using the HACMP software in conjunction with the AIX LVM advantages. Depending on the business needs, the storage subsystem alone or the entire system, including the processor, can be fully redundant.

Locality The use of separated processors and data becomes an important issue when configuring high-availability solutions. External storage solutions for RS/6000 systems can override the limitations of the 6-meter maximum length for SCSI connections. One solution is to use HACMP with geographically dispersed nodes and SSA over fiber-links.

A comprehensive list of the maximum storage capacities for various storage adapters and subsystems, high-availability configurations and performance considerations is in the sales manual pages for each of the RS/6000 models, RS/6000 storage products and storage-related publications.

The following are the descriptions for the most important disk storage subsystems for the RS/6000:

IBM 7204 External Disk Drive: This unit provides expansion storage for a single disk drive. The external disk drive capacities range between 1.1 GB and 9.1 GB. This is a practical way to expand storage when one or two extra disk drives are required beyond what will fit inside a system unit. If more than two expansion disks are required, the 7131 or 7134 should be considered.

IBM 7027 High-Capacity Storage Drawer: The 7027 provides efficient packaging of SCSI-2 F/W disk drives and should be considered where more than 30 GB of SCSI storage is required via a rack-mounted drawer. Each 7027 drawer provides up to 30 disk bays for a maximum of 67 GB of disk storage. Up to four 7027 drawers can be installed in an R00 rack system for attachment to Micro Channel-based RS/6000s. The 7027 can also accommodate tape drives, CD-ROM drives or the 24/48 GB 4 mm tape autoloader.

IBM 7131 Multi-Storage Tower Model 105: This is a low cost, stand-alone SCSI storage tower that provides storage expansion up to 63 GB using SCSI-2 F/W disk drives. This tower has additional space to accommodate tape or CD-ROM drives. Using two towers, a customer can mirror data to provide a low-cost backup solution for environments where recovery from data loss is critical.

IBM 7131 SSA Multi-Storage Tower Model 405: The IBM 7131-405 Multi-Storage Tower is an SSA tower that provides up to 45 GB of capacity. This disk subsystem should be considered for applications with performance requirements exceeding the capability of the direct SCSI-attached storage. Future expansion of the 7131 Multi-Storage Tower should provide an upgrade path to the new higher-performance IBM disk drives.

IBM 7133 SSA Disk Subsystem Models 010 and 500: The two models are functionally equivalent and can support up to 145 GB of disk storage capacity using 16 SSA disk drive units. These subsystems can be configured with redundant power for all the disk drives. Using two 7133 model 010s or model 500s, a customer can mirror data to provide a high-performance backup solution for environments where recovery from data loss is critical. These subsystems

should be considered for customers requiring high-performance disk storage and for applications with performance and maximum storage capacity requirements exceeding the capability of the direct SCSI-attached storage. These subsystems also should be considered when high-availability configurations are necessary.

IBM 7133 SSA Disk Subsystem Models 020 and 600: Combined with IBM′s family of SSA adapters, the 7133 models 020 and 600 can be configured for high performance, multiple attachments, or economical RAID 5 data-protected storage. These subsystems can provide up to 145 GB of disk storage. The SSA fiber-optic extender feature is available on all models of the 7133. The 7133 SSA disk subsystem models 020 and 600 should be considered where high performance and maximum storage capacity are necessary for applications requiring high-performance disk storage and data protection via mirroring or low-cost RAID configurations.These subsystems also should be considered when high-availability configurations are necessary.

IBM 7134 High-Density SCSI Disk Subsystem: The IBM 7134 High-Density SCSI Disk Subsystem can provide from 4 to 72 GB of disk storage in a drawer. Multiple 7134 drawers can be placed in one 7015 system rack. The 7134 should be considered (using one or more 7134 disk subsystems) where more than 50 GB of maximum SCSI differential disk drive storage capacity is required.

IBM 7135 RAIDiant Array Model 010: The 7135 RAIDiant Array model 010 is an entry-level non-RAID model. This model can be expanded to 12 disk drives, giving a maximum capacity of 54 GB. This array can be field-upgraded to a high-availability 7135 model 110 or model 210. With this array, customers can build non-protected configurations knowing that, if their requirements change, the investment in packaging and disk drive modules is protected by the ability to upgrade the subsystem.

IBM 7135 RAIDiant Array Models 110 and 210: The 7135 RAIDiant Arrays are designed to provide high-capacity disk storage, with data protection and fault-tolerant configurations. The 7135 RAIDiant arrays can be configured with two RAID controllers, in RAID levels 0, 1 and 5. These arrays can provide a maximum of 135 GB. The 7135 disk arrays should be considered for applications demanding high storage capacity and data protection (may be fault tolerant) and if performance is not the major issue.

IBM 7137 Disk Array Subsystem: The 7137 Disk Array Subsystem is a RAID storage array for those applications where fully fault-tolerant redundancy levels provided by the 7135 are not required and the storage requirements are less than 50 GB. The 7137 provides up to 62 GB of disk storage with RAID data loss protection and is a good, low-cost solution for customers with high data-availability requirements but no need for the high performance of the 7133 RAID models. Using RAID 5, all models support dynamic “hot sparing” that, in the unlikely event of a drive failure, allows automatic restoration of data redundancy without operator intervention.

Important

The maximum capacities of the external disk drives and external disk drive arrays shown above can be configured only for AIX 4.1.5. and AIX 4.2. Remember that AIX 3.2.5. does not support 9.1 GB disk drive units and has a limitation of seven devices per SCSI-2 F/W bus.

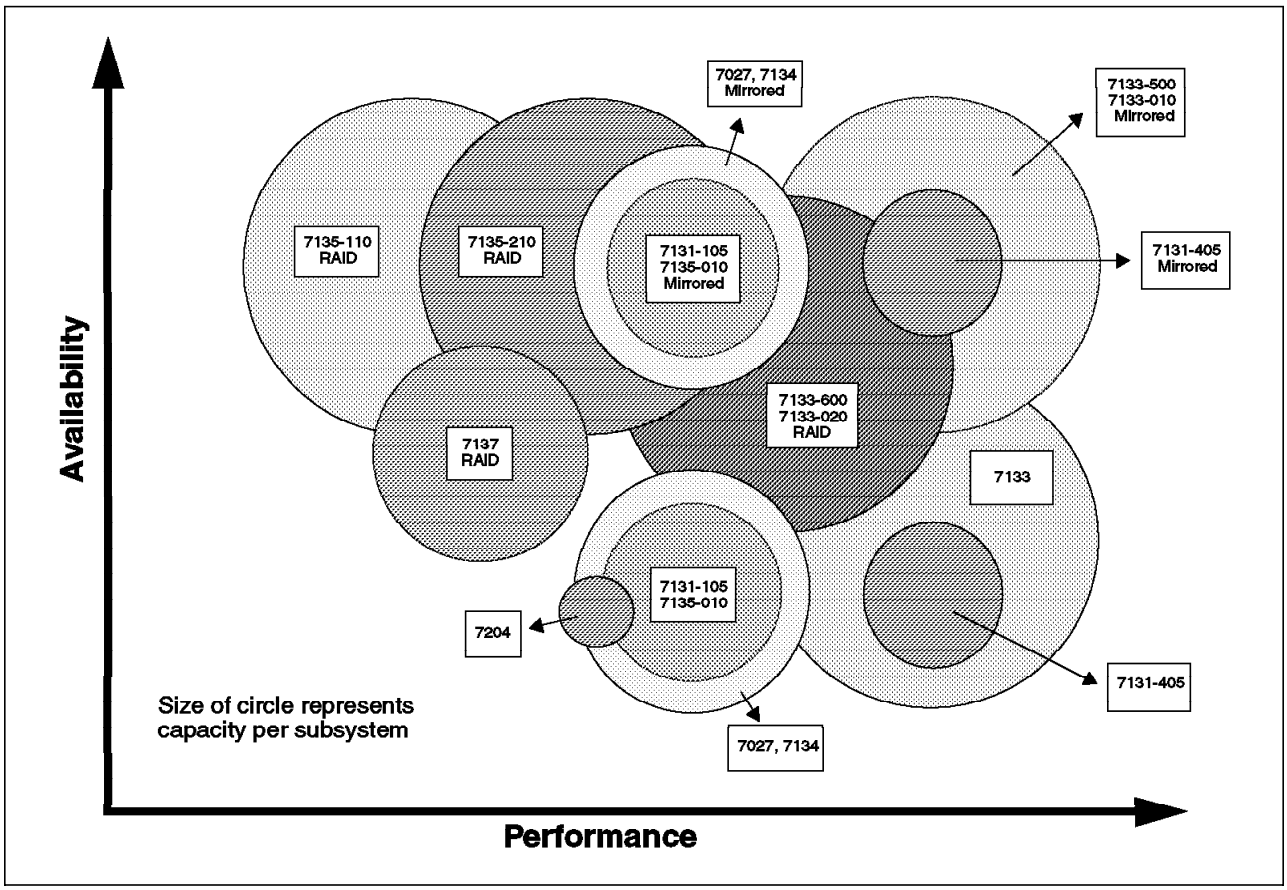


Figure 33. Disk Subsystem Positioning

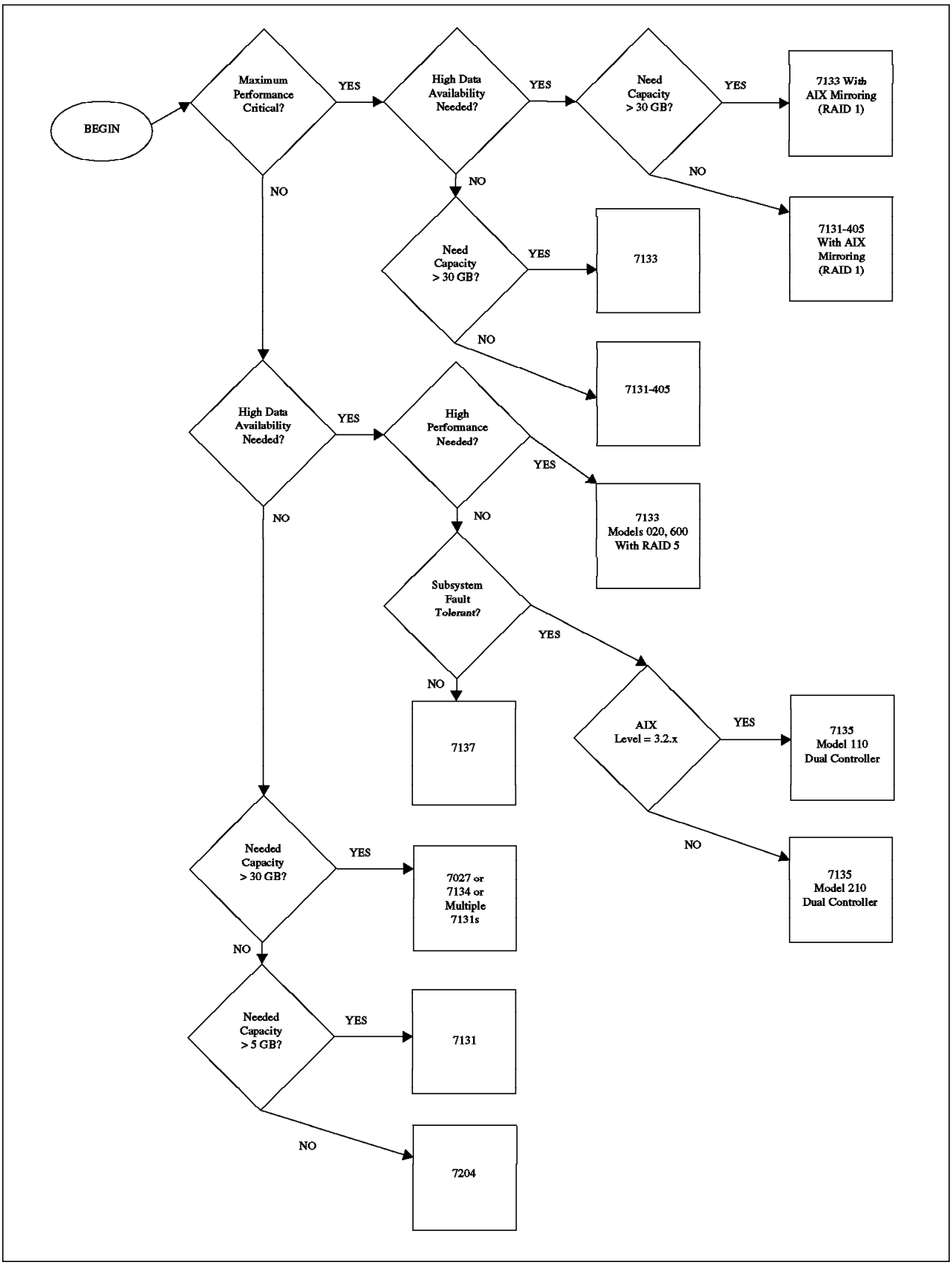


Figure 34. External Disk Storage Selection Flowchart

4.3.9 AIX Logical Volume Manager Performance and Sizing Concepts

Disk drives and disk adapters are not the only parts to be considered for I/O in an RS/6000 system. The role of the AIX logical volume manager (LVM) and the AIX File System Manager operating system components for disk storage performance and sizing must also be considered. Many of the disk array subsystem advantages cannot be exploited without an adequate configuration of the AIX storage subsystem parameters.

The AIX LVM provides many tools and capabilities to define a strategy for RS/6000 storage that is oriented toward a combination of availability, performance and cost that is appropriate for the particular site.

Information about the RS/6000 storage fundamentals, the AIX LVM and AIX operating system management can be found in *AIX Version 4 System Management Guide: Operating System and Devices* (SC23-2525) and *AIX Versions 3.2 and 4 Performance Tuning Guide* (SC23-2365).

These are the most important AIX LVM concepts that must be known before defining a storage strategy for an RS/6000 system:

- Inter disk allocation policies
- Intra disk allocation policies
- Write verify policies
- Data availability (mirroring, RAID)
- Shared storage and HA configurations (for HACMP environments)
- Logical volume reorganization
- File system defragmentation
- Map files used for precise allocation of logical volumes
- Defining raw devices for applications
- File system fragmentation
- File system compression
- File system log considerations

The following concepts about AIX performance and tuning for disk I/O are covered in *AIX Versions 3.2 and 4 Performance Tuning Guide* (SC23-2365):

- Asynchronous disk I/O servers
- Paging space distribution
- File defragmentation
- Disk I/O pacing
- Storage adapter device driver parameters
- I/O history
- Tuning sequential read ahead
- Tuning write behind

Logical Volume Striping: Striping is a technique for spreading the data in a logical volume across several disk drives in such a way that the I/O capacity of the disk drives can be used in parallel to access data on the logical volume. (The ability to create striped logical volumes is not available on AIX Version 3.2). The primary objective of striping is very high-performance reading and writing of large sequential files. Since mirroring is not supported when striping is applied, the availability of the striped logical volume is low compared with a mirrored logical volume.

Fragmented File Systems: The journaled file system fragment support allows disk space to be divided into allocation units that are smaller than the default size of 4096 bytes (in AIX Version 4 only). Smaller allocation units or "fragments" minimize wasted disk space by more efficiently storing the data in a file or directory's partial logical blocks. When a file system is created, the system administrator can specify the size of the fragments in the file system. The allowable sizes are 512, 1024, 2048, and 4096 bytes (the default).

Files smaller than a fragment are stored in a single fragment, conserving disk space, which is the primary objective. Files smaller than 4096 bytes are stored in the minimum necessary number of contiguous fragments. Files with size between 4096 bytes and 32 KB (inclusive) are stored in one or more (4 KB) full blocks and in as many fragments as are required to hold the remainder. Files that contain more than 32 KB of data are stored entirely in full blocks.

The primary performance hazard for file systems with small fragment sizes is space fragmentation. The existence of small files scattered across the logical volume can make it impossible to allocate contiguous or closely spaced blocks for a large file. The performance of accessing the large file suffers. Carried to an extreme, space fragmentation can make it impossible to allocate space for a file, even though there are many individual free fragments.

A test of writing 80 MB to a file system (using the same logical volume for fragment sizes of 512 Byte, 2 KB, and 4 KB) yields the following average response time:

Fragment Size (bytes)	Average Response Time (sec)
512	200
2028	165
4096	110

Table 11. Fragment Size/Response Time Comparison

A policy for defragmenting the space in a file system should be part of the decision to create a small fragment file system.

Compressed File Systems: The journaled file system supports compressed file systems (in AIX Version 4 only). Compressed file systems save disk space by allowing a logical block to be stored on the disk in units or "fragments" smaller than the full block size of 4096 bytes. Data compression, however, stores each logical block of an file with any size as one or more contiguous fragments. On average, data compression saves disk space by about a factor of two. The compression algorithm is a modified Lempel-Zev (LZ) algorithm.

The use of fragments and data compression does, however, increase the potential for fragmentation of the disk's free space. Fragments allocated to a logical block must be contiguous on the disk. A file system experiencing free space fragmentation may have difficulty locating enough contiguous fragments for a logical block's allocation, even though the total number of free fragments may exceed the logical block's requirements.

Because data compression is an extension of fragment support, the performance costs associated with fragments also apply to data compression. Compressed file systems also affect performance in the following ways:

- It may require a great deal of time to compress and decompress data so that the usability of a compressed file system may be limited for some user environments. Data compression increases the number of processor cycles. For software compression, the number of cycles for compression is approximately 50 cycles per byte, and for decompression 10 cycles per byte.
- Most UNIX regular files are written only once, but some are updated in place. For the latter, data compression has the additional performance cost of having to allocate 4096 bytes of disk space when a logical block is first modified, and then reallocate disk space after the logical block is written to the disk. This additional allocation activity is not necessary for regular files in a noncompressed file system.

Journaled File System (JFS) vs. Raw Devices: A raw device is an area of physical and logical disk space that is under the direct control of an application rather than under control of the operating system and file system. Applications that use character (raw) input and output, rather than block input and output of file systems, require more software overhead. For example, a lab test with an SSA storage subsystem was done with both, journaled file system (JFS) and raw device providing the data for the benchmark. Using JFS, at the highest level of performance, the benchmark I/O required nearly 100 percent of the processor resources. The equivalent benchmark using the raw device needed only 30 percent of the processor. (Note that this test was for only one application running in the system.)

In most cases, using raw devices helps improve disk I/O performance, decreasing the I/O response times by a factor between 20 and 30 percent. Bypassing the file system overhead enables applications to perform better. Raw logical volumes are most commonly used with database applications because of their need for high performance. While there is ordinarily a significant increase in performance, the actual amount of the increase depends on the database size and the driver provided by the application.

Raw devices have the following disadvantages compared to JFS:

- Data integrity
- Data security
- Data manipulation

The application should deal with these disadvantages.

Asynchronous I/O: Usually, applications use the `read()` and `write()` subroutines to access data from storage. As seen in Figure 28 on page 63 , an application using a `read()` or `write()` subroutine calls a kernel process to perform the I/O request, but it must wait until the kernel process delivers the result of the operation. An application can use the `aio_read()` and `aio_write()` subroutines to perform asynchronous disk I/O. Control returns to the application from the subroutine as soon as the request has been queued. The application can then continue processing while the disk operation is being performed.

Although the application can continue processing, a kernel process, called a server, is in charge of each request from the time it is taken off the queue until it completes. The number of servers limits the number of asynchronous disk I/O operations that can be in progress on the system simultaneously.

AIX allows the configuration of the minimum and maximum number of kernel process servers for asynchronous disk I/O requests. The maximum limits the number that can be started in response to large numbers of simultaneous requests.The result of a deficiency of servers is that disk I/O seems much slower than it should be. Not only do requests spend inordinate lengths of time in the queue, a low ratio of servers to disk drives means that the seek optimization algorithms have too few requests to work with for each drive.

4.3.10 Considerations When Configuring Storage

Before any performance improvement or sizing task can be done, the workload of the storage subsystem must be known to ensure that the performance or sizing procedures lead to the correct configuration. It is quite different to evaluate the I/O workload for an installed system than for a new system.

For installed and new systems, you must know:

- Application's I/O rate demands
- Application's disk storage demands
- Application's concurrence levels
- Database designs
- User comments about performance and expectations
- Data growth path

In addition, this information should be provided for installed systems:

- Logical volumes and file system distribution
- Hardware and software configuration of the system
- BEST/1 predict information for the I/O subsystem
- AIX disk I/O performance monitoring command output
- Trace reports for the I/O subsystem (like `filemon` reports)
- Performance Toolbox disk usage and I/O statistics

Customer Expectations: Customer expectations are of major importance when completing disk I/O performance and sizing tasks.

Transaction Files: There are thousands of files in an RS/6000 system installation. Most of these files are used once a day, once a week or never. The current workload requires only a few of the stored files in the system: generally, database and index files.

An important thing to do on every installed system is to check which of the files in the file system structure consume most of the disk I/O resources in the system. AIX provides some tools to do that. A system can be tested periodically to detect the most-used files with utilities like `filemon`. In most cases, for a reasonable time interval, 80 percent of the I/O operations in a system are made to 1-2 percent of the files, representing between 20 and 40 percent of the total disk space. Let us call these files ″transaction files″.

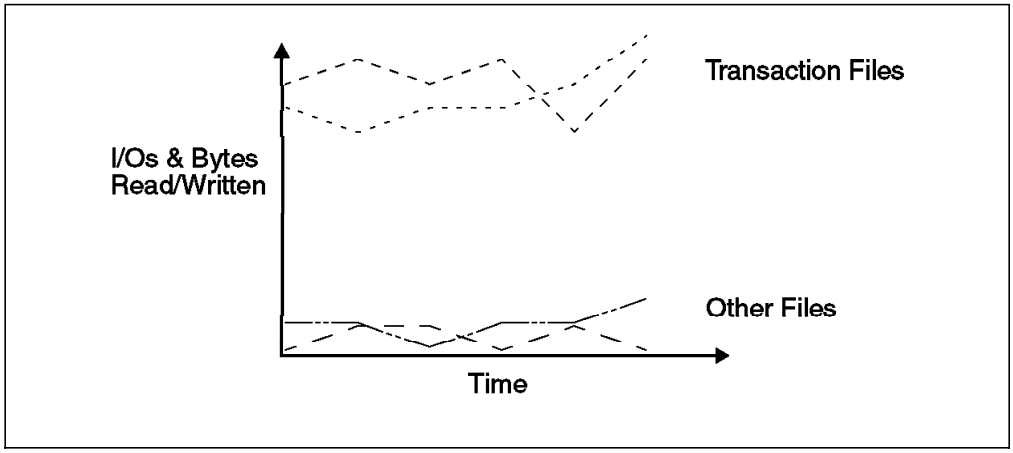


Figure 35. Transaction Files

Now, select the transaction files and assign them to high-performance disk storage subsystems in specially created file systems.

All of the following steps are necessary to ensure that the transaction file selection and storage design tasks will lead to performance improvements in the I/O subsystem:

1. Use AIX disk performance tools (like `filemon`) to check for the most-used files in the system (check for number of read/written bytes). You should select a set with no more than 30 files, but this depends on your applications and workload.
2. Ensure that all of the files in the transaction file set can be used from the user's applications via symbolic links.
3. Make a backup of all your file systems. Make a separate copy of the transaction file set using non-absolute paths. CAUTION: You should ensure that the backup media is OK before proceeding with the next steps.
4. If necessary, install high-performance disk storage subsystems for the transaction file set.
5. Delete all your file systems from the system's file system set.
6. Create a file system with sufficient space for the transaction file set. Ensure that the transaction file set fits in the file system with sufficient space for growth. This file system should be created following the recommendations for high-performance file systems, described in the *AIX Version 4 System Management Guide: Operating System and Devices* (SC23-2525).
7. Restore the transaction file set into the high-performance file system.
8. Create all the other file systems following an I/O access priority order.
9. Restore all other files except the transaction files in their respective file systems.
10. Create symbolic links for each of the files in the transaction file set from their original paths to the new ones.

The high-performance file system should have the following characteristics:

Intra physical policy	**Center**
Inter physical policy	**Maximum**
Scheduling policy	**Parallel**

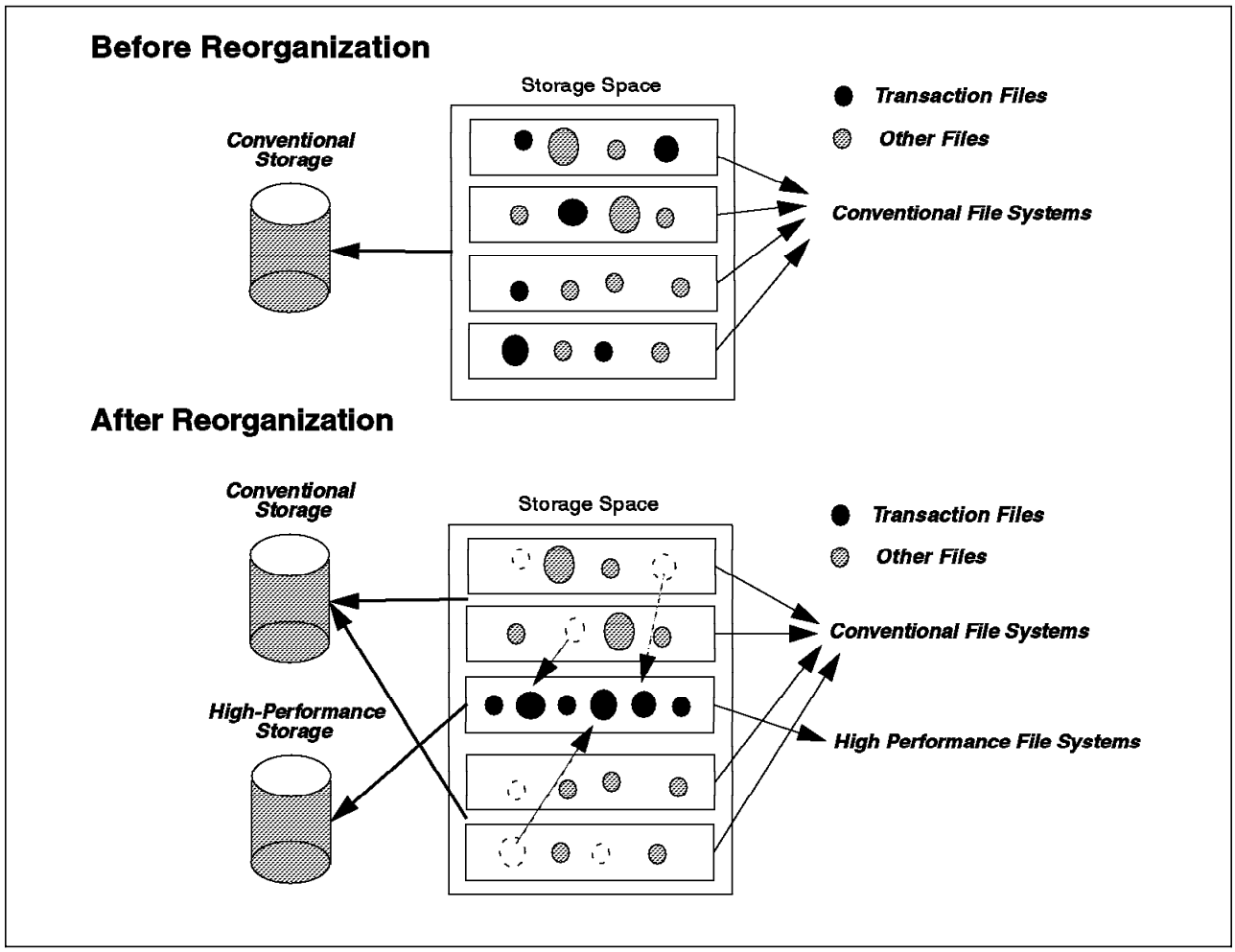


Figure 36. Transaction File Optimization

General Considerations: The following are some general considerations when configuring disk storage subsystems:

- It is best not to use more than 40 percent of a shared disk's rated capacity; a higher utilization may demand high processor time and can cause a disk bottleneck.
- A Micro Channel bus can only support four SCSI-2 differential adapters (feature code 2420).
- Try to connect no more than four disk drives per SCSI or SCSI-2 adapter. If you have high-capacity SCSI disk drives and the workload demands high throughput in the I/O subsystem, you may not want to connect more than two or three disks per SCSI or SCSI-2 adapter.
- Although some SCSI-2 F/W adapters can support up to 45 devices, only 15 can share the same bus. Whenever possible, try to balance the quantity of disk drives between the SCSI buses in a SCSI-2 F/W adapter.
- Always configure the maximum possible cache for RAID units. The only case where this is not valid is for situations when data availability is necessary but not performance.
- Always think about SSA where applications demand high performance from the disk storage subsystem.

- Think about SSA where applications have fast-growing disk I/O requirements.
- Whenever you have SCSI disks installed and data is growing quickly, think about SSA to allocate the transaction files and free SCSI storage space.
- It is better to use mirroring rather than RAID for high-demand files (transaction files). Remember that, for massive update operations, RAID 5 suffers a write penalty.
- If high availability is required, then SCSI-2 differential or SSA external subsystems should be configured.
- If you need to connect a large number of existing SCSI disks to an RS/6000 system, you might consider using bridge controllers to attach the disks to SSA adapters in the RS/6000. These hybrid subsystems eliminate the cable length and device number limitations of the SCSI buses. Be aware, though, that hybrid subsystems can have limitations in performance or fault tolerance compared to native SSA implementations.

If ***[Total storage capacity needed > 10 GB]*** *then*

Perform transaction files selection as described above

TFSize = Sum of all the file sizes in the transaction file set

OFSize = Sum of all the file sizes in the other-than-transaction-file set

AllStorage = TFSize + OFSize

If ***[(TFSize < 70% of AllStorage]*** *then*

{Because the disk storage needs for other than transaction files are significant, conventional disk storage should be considered for other than transaction files}

Use the **"External Disk Storage Selection"** *flowchart to select the appropriate disk storage subsystem for the other-than-transaction-file set*

If ***[(TFSize > 20% of AllStorage] or [high performance in I/O operations for the transaction file set is CRITICAL]*** *then*

Use the **"External Disk Storage Selection"** *flowchart to select the appropriate high-performance disk storage subsystem for the transaction file set*

Otherwise

{Because the disk storage needs for transaction files are not significant, high-performance disk storage should not be considered}

Use the **"External Disk Storage Selection"** *flowchart to select the appropriate disk storage subsystem for the transaction file set*

EndIf

Otherwise

{Because the disk storage needs for other than transaction files are not significant, conventional disk storage should not be considered}

Use the **"External Disk Storage Selection"** *flowchart to select the appropriate high-performance disk storage subsystem for both transaction file set and other-than-transaction-file set*

EndIf

Otherwise

Use the **"External Disk Storage Selection"** *flowchart to select the appropriate disk storage subsystem for the storage capacity needed*

EndIf

Figure 37. External Disk Storage Selection Algorithm

4.3.11 References

AIX Version 4 System Management Guide: Operating System and Devices, SC23-2525

AIX Versions 3.2 and 4 Performance Tuning Guide, SC23-2365

IBM StorageSmart Solutions Marketing Guide For RS/6000 And Selected Unix Systems. Package: RS6STORG on MKTTOOLS

IBM 7133 Disk Subsystem. Package: 7133PUBS on MKTTOOLS

`http://www.storage.ibm.com`

4.4 Asynchronous Communication Adapters

Serial asynchronous communication was the first commercial communication method for the UNIX environment.

Based on the teletype communication method, serial asynchronous ports permit the connection of several serial devices such as terminals, printers, fax machines, and modems.

Serial asynchronous communication is associated with a hardware line between the devices and the host. To understand how it works, we will give a brief overview of serial communications.

Inside the CPU, data and addresses are processed as words (8 bits, 16 bits, 32 bits, etc.). It uses a parallel configuration of data bits, and the number of wires must be equal to the number of parallel processed data bits.

For serial communication, only one wire is required to send data to the devices. There is a converter to change a sequential stream of data into a parallel stream of data, and the parallel data stream is sent by the CPU as a serial stream. This converter is called the serial communication adapter.

These streams need a signal to define when the data starts and when it ends. The synchronization is the process of timing the serial transmission in order to properly identify the data being sent. The two most common modes of synchronization are: synchronous and asynchronous.

Synchronous communication is used for a continuous transfer of large amounts of data. The data blocks are grouped and spaced in regular intervals and are preceded by special characters called sync or synchronous idle characters. The *sync* character is used to synchronize the connection so data transmission can begin.

Asynchronous communication is used when the data transfer is randomly started and stopped. In asynchronous transfers, there is a start bit and stop bit that specify the beginning and the end of a character. Each character is preceded by a start bit and followed by one or more stop bits.

4.4.1 Terms Used in Serial Communication

Bits per Character: It indicates the number of bits used to represent a single data character during serial communication. With seven bits, it is possible to represent 128 characters that make up the standard ASCII character set. With eight bits, it is possible to represent 256 characters that make up the ASCII extended character set.

Bits per Second (bps): It is the number of data bits (1s or 0s) that are transmitted per second over the communication line. It refers to the communication line speed.

Baud Rate: It is the number of times a serial communication signal changes states per second. If the signal changes each time a data bit enters the device, then bps is equal to baud. This is the most common case in the serial asynchronous communication because the data transmission is made over a baseband modulation (a low voltage for a 1, a high voltage for a 0).

Parity Bit: It is an optional parameter used in serial communication to determine if the data character being transmitted is correctly received by the remote device. Options are: none, even, odd, space, and mark. It is used for error detection on a communication line. Both sending and receiving systems must be configured identically.

RS-232-C, RS-232-D and RS-422 Standard: It defines the mechanical and electrical specifications for the most common connector designs in serial data communication. It defines 25 pins with assigned signals. With RS-232-C, devices are divided in two types: DTE (data terminal equipment) such as computers and terminals, uses pin 2 (TxD) as an output, and DCE (data communication equipment) such as modems, uses pin 2 as input. The RS-232-D is a revision of RS-232-C. The TxD is the transmit data pin, and the RxD is the receive data pin.

The RS-422 uses two pairs of cables to provide a differential signal for TxD and RxD. This differential signal makes the serial communication more immune to electrical interference, meaning the RS-422 can provide longer serial communication cables than RS-232.

4.4.2 Communication Methods

Simplex: It is the simplest connection between two devices. Simplex, or one-way communication, allows data to be transmitted in one direction only and requires only two lines to be connected: TxD (or RxD) and the signal ground (SG).

Duplex: There are two forms of two-way communication: half duplex and full duplex.

- **Half duplex.** In this mode, using a single pair of wires allows data to be transmitted in two directions but not simultaneously.
- **Full duplex.** In this mode, data communication can take place in two directions simultaneously over two separate lines or wires.

4.4.3 Flow Control

Serial devices do not process data as quickly or efficiently as the CPU they are connected to. There are memory buffers associated with the data transmission that need some type of flow control to limit the amount of data transmitted by or to the CPU.

Flow control also is referenced as handshaking. There are two types of flow control: hardware and software.

With hardware flow control, wires and voltage levels are used for data transmission control.

- **RTS/CTS** (ready to send/clear to send). It uses pins 4 and 5 to control the flow. A low RTS signals the sending site to stop transmitting data. When the buffer is almost empty, RTS is raised again, issuing a signal to send more data to the other site.
- **DTR/DSR** (data terminal ready/data set ready). This hardware flow control is normally generated by devices, such as printers. DTR indicates that the device is ready to communicate with the CPU. The CPU uses DSR to control the data flow.

- **DCD** (data carrier detect). It is a signal in pin 8 also referred to as received line signal detector. DCD is an output signal for a modem and an input signal for the CPU. When DCD is high, the CPU knows a modem connection has been made. With DCD in low state, the CPU knows the modem connection is terminated.

The software flow control involves sending of data transmission control characters along the data stream.

- **XON/XOFF** (transmitter on/off). This flow control operates with the buffer capacity in both sites of the data transmission. Just before the device buffer reaches its maximum capacity, the device will send an XOFF character to the CPU. The data transmission is then stopped. When the device buffer is almost empty, it will send an XON signal character back to the CPU, causing more data to be sent.

4.4.4 IBM RS/6000 Asynchronous Hardware

From a hardware perspective, IBM offers 8-port and 128-port asynchronous adapters to connect asynchronous devices such as terminals, modems, and printers to RS/6000 machines. The Serial Communications Network Servers (7318-P10 and 7318-S20) provide RS-232/422 asynchronous ports and connect to RS/6000 machines through an Ethernet LAN connection.

IBM RS/6000 serial asynchronous adapters can be classified into two categories: intelligent and non-intelligent serial adapters.

- The non-intelligent adapters rely on the system's CPU for handling the I/O buffering for all the serial ports. This handling is made through an interrupt to the CPU that is sent every time a communication takes place. Normally, these interrupts are generated when a character is transmitted or received on the port or when a flow control condition occurs. An interrupt means the CPU must stop what it is doing and serve that request.
- The intelligent adapters have their own processor, memory, and buffer RAM. The adapter does much of the serial I/O work normally handled by the CPU. It will make up a package using its buffers, and it interrupts the CPU only when its buffers are full.

The eight-port asynchronous adapters (MCA and ISA type) are non-intelligent adapters. The adapter connects to the eight-port RS-232/422 interface unit through a connecting cable.

The 128-port asynchronous adapters (MCA and ISA type) are intelligent adapters. One 128-port asynchronous adapter can connect up to eight remote asynchronous node (RAN) units, where each of these units provides 16 RS-232/422 interface ports to be connected to serial devices.

Serial Communication Network Server: The Serial Communication Network Server (7318) is an alternative to the asynchronous communication adapter. Two models, the P10 and S20, provide a 16-port RS-232/422 and one parallel Centronix interface. They connect to the RS/6000 machine through an Ethernet cable connection. The P10 model uses the IPX/SPX protocol, and the S20 uses IPX/SPX or/and TCP/IP. The network terminal server (NTS) converts asynchronous communication I/O requests into an rlogin or telnet session for the interactive terminal session.

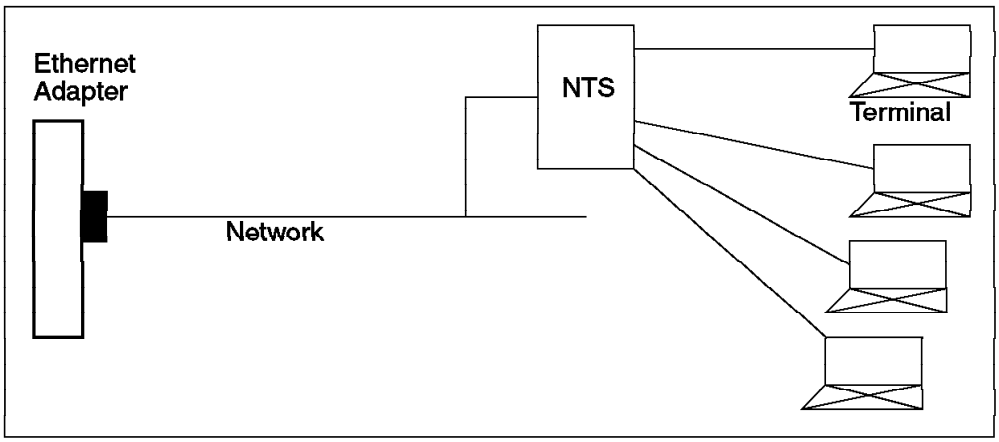


Figure 38. Serial Communication Network Server Environment

The network server offers simultaneous access to multiple hosts. Because the terminal sessions are rlogin or telnet type sessions, a single station connected to an NTS port can have up to seven concurrently open sessions, including an additional parallel port for high-speed printing.

Additional performance improvements are achieved with the network terminal accelerator (NTX) adapter because the NTX will offload the rlogin and telnet daemon processes from the host system to the adapter.

The NTS with or without NTX will have a different performance impact on the host CPU because NTS will be more related to the LAN performance.

4.4.5 Asynchronous Adapter Considerations

Some considerations are necessary to understand the performance of the asynchronous adapter.

AIX supports two processing modes for the asynchronous adapter: raw mode and cooked mode. In the cooked mode, the asynchronous device driver and AIX software (e.g. login) will process any input or output characters depending on the processing characteristics defined (ipost, opost). In the raw mode (i.e connecting to fax, printer), any input or output characters will not be processed or checked by the asynchronous device driver. The mode depends on the type of applications or processes that will be run over the asynchronous adapter and ports.

The baud rate of an asynchronous port will have an impact on the CPU utilization and, therefore, will determine the number of asynchronous ports effectively supported for performance considerations. For more detailed information about this, please refer to *AIX Versions 3.2 and 4 Performance Tuning Guide* (SC23-2365).

The 128-port asynchronous adapter can be used to support many asynchronous ports and should be considered especially for RS/6000 machines that do not have many I/O slots. Another alternative is to use NTS because the network connection can support many connections through the LAN adapters. This is very important, especially on a system designed for Internet dial-in connections.

4.4.6 Conclusion

The performance of the 128-port adapter and NTS solutions shows that the asynchronous environment is still a good solution for multiuser systems. Also, the cost of the implementation is lower than that of other solutions.

NTS has good integration with the workstation environment through structured cabling for a heterogeneous network, share devices, remote access, and higher security.

4.4.7 References

AIX Versions 3.2 and 4 Performance Tuning Guide, SC23-2365

http://www.rs6000.austin.ibm.com/hardware

4.5 LAN / WAN Adapters

In growing network centric computing, client/server, and workgroup environments, local area network (LAN) and wide area network (WAN) performance becomes more and more important. Network performance is dependent on the type of the network, such as token ring, Ethernet, FDDI, or ATM. But it is also highly dependent on the application, the frequency of data transfers, and the amount of data that is transferred through the network, as well as on the design of the entire network.

One basic thing to understand is that you should never expect network traffic to be as fast as the indicated throughput of the adapter. Throughput can be defined as the amount of data exchanged between systems over a given time interval. In a real production environment, individual components within the larger network can also affect throughput. In fact, the slowest component within a network is the bottleneck that determines that network's maximum throughput.

The indicated adapter throughput is defined by the clock rate of the adapter. (For instance, the 100 Mb/s ATM adapter has a clock rate of 100 MHz.) For the indicated throughput, only the adapter is examined but never environmental factors such as the system bus usage (other adapters are using the system bus at the same time), active applications, users and traffic on the network. Network throughput, measured by the AIX Performance Tools, might also lead to misunderstandings. The tools consider only raw data throughput, not the protocol header. For instance, for a 100 Mb/s ATM adapter, you will see a maximum throughput of about 91 Mb/s.

This chapter will describe the different kinds of LAN and WAN protocols, the adapters and their related performance features. It also offers some considerations about performance tuning.

4.5.1 Ethernet

Ethernet is a logical communication protocol that permits multiple devices to talk to each other over a variety of physical media. This logical protocol is called carrier sense multiple access with collision detection (CSMA/CD). CSMA/CD is a broadcast mechanism where one station speaks while all others listen. When two stations are sending data at the same time, a collision takes place. With CSMA/CD, both stations detect the collision, back off, and retry later. Ethernet is

a shared network (this means all the stations share the same bus), and it is for the most part a 10 Mb baseband LAN.

AIX provides two Ethernet interfaces, standard Ethernet and IEEE 802.3 Ethernet (IEEE - Institute of Electrical and Electronics Engineers). The frame formats for Ethernet and IEEE 802.3 are not the same. However, both protocols use the same medium and access method. LAN stations running these protocols can share a common bus, but they cannot communicate with each other.

Hardware: There are Ethernet adapters available for all three kinds of RS/6000 buses: Micro Channel architecture (MCA), peripheral component interconnect (PCI) and industry standard architecture (ISA). Some RS/6000 models have integrated Ethernet adapters. The adapters differ by the variety of Ethernet cabling:

- **10Base-2** (10 Mb/s, thin Ethernet or BNC, maximum segment length 185 meters, maximum 30 stations per segment) uses inexpensive coaxial cable as its transmission medium. It depends upon transceiver logic in the host adapter to resolve collisions.
- **10Base-5** (10 Mb/s, thick Ethernet or DIX, maximum segment length 500 meters, maximum 100 stations per segment) uses a special double-shielded coaxial cable as its transmission medium. It depends on an external transceiver to resolve collisions.
- **10Base-T** (10 Mb/s, twisted pair Ethernet or AUI, uses star topology (different from coaxial types of Ethernet), maximum cable length 100 meters between any two end points in the network, maximum number of workstations limited by the number of connections in the hub) uses unshielded twisted pair (UTP) cables as its transmission medium. It depends upon an active hub to resolve collisions between stations trying to talk at the same time.

Advantages: Ethernet is the most common and widely used LAN type. It is cheap and relatively easy to handle. UTP Ethernet bears the advantage that the cabling can be used flexibly, and eventually, an existing cabling in the location (such as telephone cabling) could be used.

Disadvantages: Coaxial Ethernet is very susceptible to interference. If, for instance, a cable has been disconnected but not terminated, the whole segment is not usable anymore, and the system administrator has to find where the error occurred. Also, UTP cabling (used for 10Base-T) is more susceptible to interference than shielded twisted pair.

Architectural constraints (maximum segment lengths) derive from timing parameters of the CSMA/CD protocol. For the collision detection and recovery mechanism to operate properly, the time interval between when two LAN adapters recognize a collision is limited. As a result, the overall size of the system is a factor of the propagation time of the signals over the longest path. The more systems in an Ethernet, the more collisions that will occur. In a busy Ethernet, the collisions can become a real performance burden.

Ethernet Performance Tuning Recommendations: Ethernet is one of the contributors to the ″least common denominator″ algorithm of maximum transmission unit (MTU) choice. If a configuration includes Ethernet and other LANs and there is extensive traffic among them, the MTUs of all of the LANs may need to be set to 1500 bytes to avoid fragmentation when data enters an Ethernet.

- The default (and maximum) MTU of 1500 bytes should not be changed.
- Applications should use block sizes in multiples of 4096 bytes.
- Socket space settings can be left at the default values.
- If the workload includes extensive use of services that use UDP, such as NFS or RPC, *sb_max* should be increased to allow for the fact that each 1500-byte MTU uses a 4096-byte buffer.

Conclusion: Ethernet might be used in cases where the network is relatively small, an inexpensive solution is desired or flexible usage of the medium (cabling) is required. It should be considered that the capacity is restricted and that Ethernet is relatively susceptible to interference.

4.5.2 Token Ring

The token ring model is a type of LAN that was developed under the auspices of the IEEE 802.5 Subcommittee. It is a token access procedure used with a sequential topology (ring). All stations in the ring can receive the token, but only one station, the one that holds the token, can transmit at a time. This avoids the possibility of data collisions because there is only one token. Ring speed in a token ring can be either 4 or 16 Mb/s.

Hardware: RS/6000 token ring adapters are available for Micro Channel, PCI and ISA buses. They are all IEEE 802.5 compatible and support both 4 and 16 Mb/s networks. Depending on the type of adapter, the ring speed is set by autosense or program.

The IBM token ring network implementation uses shielded twisted pair (STP) cabling.

Advantages: Token ring is a convenient, easy-to-handle solution to connect machines in a LAN. It provides more capacity than conventional Ethernet, and due to its architecture and the STP cabling, token ring allows more-flexible designs that are less susceptible to interference. Because of the sequential topology, even busy networks are safe from collisions, which can take a heavy toll on Ethernet performance.

Disadvantages: Token ring is relatively expensive due to the STP cabling and the required multistation access unit (MAU) to connect the stations. The network's capacity is limited to 16 Mb/s.

Token Ring (4 Mb) Performance Tuning Recommendations: The default MTU of 1492 bytes is appropriate for Token Rings that interconnect to Ethernet or to heterogeneous networks in which the minimum MTU is not known.

- Unless the LAN has extensive traffic to outside networks, the MTU should be raised to the maximum of 3900 bytes.
- Applications should use block sizes in multiples of 4096 bytes.
- Socket space settings can be left at the default values.
- If the workload includes extensive use of services that use UDP, such as NFS or RPC, *sb_max* should be increased to allow for the fact that each 1492-byte MTU uses a 4096-byte buffer.

Token Ring (16 Mb) Performance Tuning Recommendations: The default MTU of 1492 bytes is appropriate for token rings that interconnect to Ethernet or to heterogeneous networks in which the minimum MTU is not known.

- Unless the LAN has extensive traffic to outside networks, the MTU should be increased to 8500 bytes. This allows NFS 8 KB packets to fit in one MTU. Further increasing the MTU to the maximum of 17000 bytes seldom results in corresponding throughput improvement.
- Applications should use block sizes in multiples of 4096 bytes.
- Socket space settings can be left at the default values.
- If the workload includes extensive use of services that use UDP, such as NFS or RPC, the MTU must be left at the default because of interconnections, and *sb_max* should be increased to allow for the fact that each 1492-byte MTU uses a 4096-byte buffer.

Conclusion: Token ring is preferable in LAN environments requiring more reliability, easy handling and a greater capacity than the 10 Mb/s Ethernet limit.

4.5.3 FDDI

The fiber distributed data interface (FDDI) is an ANSI/ISO standard for a high-speed 100 Mb/s optical LAN interface. The general purpose of FDDI is the interconnection of computers, networks and peripheral equipment using optical fiber cable in a dual, counter-rotating ring configuration. In the wake of a failure on the primary ring, the secondary ring can become active as a backup ring without user intervention or interruption.

FDDI can connect as many as 500 stations with a maximum link-to-link distance of two kilometers and a total LAN circumference of 100 kilometers. Like token ring, FDDI is a shared network that uses a sequential (ring) topology.

Hardware: RS/6000 FDDI adapters support optical cabling for a line speed of up to 100 Mb/s. FDDI adapters are supported on RS/6000 Micro Channel systems.

An RS/6000 FDDI adapter can be ordered as a single ring adapter (in order to connect the system to a single FDDI ring), or an additional dual ring upgrade kit can be ordered to connect the system to a dual FDDI ring.

Advantages: Even though FDDI uses the same topology as token ring, it provides incomparably more capacity. Fiber-optic cable is designed to be immune to electromagnetic interference. Therefore, using fiber technology may prevent jamming by high power emissions. Additionally, fiber-optics do not emit electromagnetic radiation. This feature enhances security, since it helps prevent eavesdropping. Due to the dual ring concept, FDDI provides a fundamental functionality to maintain the reliability of the LAN.

Disadvantages: Because of the fiber-optic cabling, FDDI networks are quite expensive. Even though FDDI provides a high capacity, it is implemented as a shared network concept, which means that all the stations have to share the available bandwidth. Therefore, FDDI is not suitable for carrying audio and video data.

FDDI networks provide a high capacity, and the system CPU usage required to handle this data can be quite extensive.

FDDI Performance Tuning Recommendations: Despite the comparatively low MTU, this high-speed medium benefits from substantial increases in socket buffer size.

- Unless the LAN has extensive traffic to outside networks, the default MTU of 4352 bytes should be retained.
- Where possible, an application using TCP should write multiples of 4096 bytes at a time (preferably 8 KB or 16 KB) for maximum throughput.
- Use `no -o *_*space=NewSize` to set the TCP and UDP socket send and receive space defaults to `NewSize` bytes. `NewSize` should be at least 57344 bytes (56 KB).
- Use `no -o sb_max=(2*NewSize)` to raise the ceiling on socket buffer space.
- For RS/6000 Model *90 or faster, use `no -o rfc1323=1` to allow socket buffer sizes to be set to more than 64 KB. Then use the previous procedure with `NewSize` of at least 128 KB.

Conclusion: Most likely, FDDI will be used as a backbone to tie together other LANs such as token ring or Ethernet LANs. It provides a high reliability and security as well as a strong capacity. Nonetheless, the expenses are high.

4.5.4 ATM

Asynchronous transfer mode (ATM) is a switching technology based on 53-byte, fixed-length cell calls. It has very high-speed capabilities (100 Mb/s and 155 Mb/s) and is suitable for carrying data, voice, and video in a LAN or WAN environment.

Hardware: The two standardized ATM adapters presently supported on the RS/6000 are both Micro Channel adapters.

- Turboways 100 ATM adapter/A FC #2984. Supported in models 3xx, 5xx and 9xx only.
- Turboways 155 ATM adapter (Type 9-9) FC #2989.

A fiber-optic cable is used to set up a back-to-back connection between two RS/6000 systems (without ATM switch) or to set up a connection to a single ATM switch. The Turboways ATM implementation supports 1024 point-to-point and point-to-multipoint virtual connections.

There is a limit of two ATM adapters on the same bus.

Advantages: The main reason, ATM is so fast is that the data to be transferred is split into equal-sized 53-byte cells. These are relatively small units that can be handled by the ATM hardware instead of software.

In numbers:

- At a write size of 1 to 1024 bytes, ATM can be 10-20 percent faster than FDDI.
- At a write size of 4 K to 16 K, ATM can be 5-10 percent faster than FDDI.

Another ATM advantage is that it is not very CPU intensive. (FDDI may consume about 40 to 57 percent more CPU time using 100 Mb/s adapter.) One of the reasons for this effect is that the Turboways ATM adapter uses buffers in real memory for data communication between the application and the device driver. One more reason for low CPU usage is that the Turboways ATM adapters have

an onboard i960 co-processor, which minimizes demand on the CPU. Because ATM is a switched network, the full bandwidth is provided for each exclusive connection between two stations in the network.

Disadvantages: Right now, ATM should not be a bottleneck. Before ATM starts becoming a bottleneck, other resources in the system (like CPU, system bus or disks) usually will become a bottleneck. However, when several systems are sending data to a host at the same time, they will have to share the available bandwidth of the adapter. Another disadvantage is that ATM does not support broadcasts. Therefore, any application that relies on broadcasts will not work over ATM.

ATM Performance Tuning Recommendations

- Unless the LAN has extensive traffic to outside networks, the default MTU of 9180 bytes should be retained.
- Where possible, an application using TCP should write multiples of 4096 bytes at a time (preferably 8 KB or 16 KB) for maximum throughput.
- Use `no -o *_*space=`*NewSize* to set the TCP and UDP socket send and receive space defaults to `NewSize` bytes. `NewSize` should be at least 57344 bytes (56 KB).
- Use `no -o sb_max=(2*`*NewSize*`)` to raise the ceiling on socket buffer space.
- For RISC System/6000 Model *90 or faster, use `no -o rfc1323=1` to allow socket buffer sizes to be set to more than 64 KB. Then use the previous procedure with a `NewSize` of at least 128 KB.

Conclusion: Whenever an application requires a strong, powerful and reliable network, like for the transfer of video and audio, ATM should be taken into consideration.

4.5.5 X.25

X.25 provides the ability to transmit data between remote machines in a WAN. It is a set of recommendations (the 25th in the X series) from the International Telegraph and Telephone Consultative Committee (CCITT). These recommendations define a standard network access protocol for attaching different types of computer equipment (such as data terminal equipment - DTE) to a packet switching data network (PSDN). This protocol is particularly useful for communication between different types of computer systems and for accessing public databases. Therefore, X.25 is not an end-to-end protocol. It can be used to provide a network service for higher-level protocols, such as system network architecture (SNA) and transmission control protocol/internet protocol (TCP/IP).

The X.25 implementation supports four physical interfaces: V.24, V.35, V.36 and X.21bis.

Hardware: The IBM X.25 adapters support three different line driver circuits to accommodate three different types of connections: X.21, X.21bis/V.24 and X.21bis/V.35. These three circuits are all connected to the X.25 co-processor′s 37-pin connector or to the Portmaster adapter′s electrical interface board (fanout connector). IBM X.25 adapters are provided for Micro Channel- and ISA-based RS/6000 systems. There is only one co-processor for the RS/6000 that supports V.36: the ARTIC960 Co-Processor, 6-Port V.36.

Advantages: The most important advantage of X.25 is that it is basically available all over the world. Therefore, X.25 can be a cost-effective means of networking systems in a wide geographical area, compared to traditional dial-up (circuit switched) connections or remote bridged LANs connected by leased lines. It provides worldwide interconnection for international corporations.

Disadvantages: X.25 can be slow because data communication is not handled sequentially. Individual data packets might take different routes and may arrive out of order at the receiving system. This means that data transfer is as fast as the slowest packet. In addition, X.25 requires a reassembling of the data on the receiving system as well as error correction. Because network speed is under the control of the network provider, the speed of the data transmission is unpredictable.

X.25 Performance Tuning Recommendations: When you customize your network parameters, consider the following points:

- Since there is overhead with every packet, maximize the packet size and send full packets whenever possible. Also, try to eliminate unnecessary protocol-only packets for the higher-level protocols like TCP/IP and SNA.
- Do not use D-bit acknowledgment because it generates an acknowledgment for each packet sent.
- Maximize frame window size. Keep in mind that when a packet is lost, all the packets of the window where the loss has occurred will be sent again. If your network is reliable, you can set a big window size.
- Each system in the network should have the same MTU size.
- MTU size and X.25 packet size should be equalized.

Conclusion: X.25 might be used for an inexpensive WAN connection and in cases where there is no possibility to connect remote systems via another protocol.

> For more detailed information on X.25, refer to the *AIX/6000 X.25 LPP Cookbook* (SG24-4475).

4.5.6 ISDN

Integrated-services digital network (ISDN) allows RS/6000 systems to connect to public or private networks.

Hardware: There are no IBM ISDN adapters available at the moment.

Advantages: Unlike X.25, ISDN is an end-to-end protocol. This allows data to be transmitted sequentially, which minimizes the cost of error correction time.

Disadvantages: There are no IBM adapters available.

ISDN Performance Tuning Recommendations: It will be best to ask the ISDN hardware provider in your country for tuning recommendations.

4.5.7 HIPPI

The IBM high performance parallel interface (HIPPI) is an 800 Mb/s (= 100 MB/s) data transfer device that operates in a variety of high-speed computer environments. It is an implementation of the ANSI X3.183-1991 high performance parallel interface (HIPPI-PH) standard.

Hardware: The IBM HIPPI can attach an RS/6000 Micro Channel system to devices that comply with the ANSI standard such as other RS/6000 systems, selected members of the IBM ES/3090 or IBM ES/9000 family of computers, other vendors' supercomputers, high-performance disk arrays and tape subsystems.

There is one available Micro Channel HIPPI adapter for the RS/6000:

- High Performance Parallel Interface (HIPPI) adapter FC #2735.

Advantage: HIPPI is able to transfer data between two systems at one of the industry's highest capacities and speeds. A sustained data transfer rate up to 66 MB/s between two RS/6000 systems to the user application software layer is possible.

Disadvantage: The price level for HIPPI is quite high. HIPPI can only be used locally either point to point or through a switch. The HIPPI adapter takes up to five Micro Channel slots. The maximum supported connectivity length is 25 meters by copper cable.

HIPPI Performance Tuning Recommendations

- The default MTU of 65536 bytes should not be changed.
- Where possible, an application using TCP should write 65536 bytes at a time for maximum throughput.
- Set *sb_max* to a value greater than 2*65536.
- TCP and UDP socket send and receive space defaults should be set to 65536 bytes. Use `no -o rfc1323=1` to allow socket buffer sizes to be set to more than 64 KB.

Conclusion: Due to its high capacity, HIPPI is suitable for companies requiring large and high-speed file transfer between two systems.

4.5.8 Fiber Channel

The IBM fiber channel is a high-speed, long distance channel, compliant to the ANSI X3.230-1994 standard. It offers high-performance CPU-to-CPU and CPU-to-I/O connectivity. It provides a capacity of 1 Gb/s over a distance up to 10 km for either point-to-point connection or device-to-fiber channel switch connection.

Hardware: There is one Micro Channel adapter for the RS/6000 available.

- Fiber channel/266 adapter - Type 8-X - FC #1906.

This adapter is not supported on SMP systems.

Advantages: The fiber channel adapter provides a high-performance connection between systems that support TCP/IP via fiber channel. Selected RS/6000 systems, RS/6000 SP systems, and disk arrays can be used with it.

Disadvantages: The range of attachable devices is limited to a few products within the RS/6000 system family.

Conclusion: The IBM fiber channel might be chosen if RS/6000 systems, transferring large amounts of data, must be connected. The fiber channel technology is also very suitable in combination with other IBM products such as ATM, AIX Version 4 mirroring, SSA (serial storage architecture) or Geo-HA (geographical high availability).

4.5.9 BMPX

The block multiplexor channel (BMPX) is a parallel channel to connect RS/6000 servers with System/370 systems using a copper bus and tag cabling up to a distance of 122 meters at a channel speed of up to 4.5 MB/s.

Hardware: IBM S/370 Block Multiplexer Channel adapter FC #2755 and the S/370 Channel Emulator/A Type 5-4 FC #2759 are Micro Channel adapters for RS/6000 servers.

Advantages: BMPX provides good performance at a low price level on the RS/6000 side. It provides features (CLIO/S = Client Input/Output Sockets) to replace slow TCP/IP applications (such as FTP) that allow data to be sent at a high speed on channel level.

Disadvantages: BMPX is relatively slow and restricted in the maximum distance between the systems. Two adapters are supported, one for 3088 emulation and one for peripherals.

Conclusion: BMPX is actually superseded by ESCON.

4.5.10 ESCON

The enterprise systems connection architecture (ESCON) allows RS/6000 servers to have a peer-to-peer, processor-level communication with ES/9000 systems. It is a fiber-optic, high-speed channel connection that provides a channel speed of up to 17 MB/s on selected models of the 9021 processors and a multimode fiber support up to 3 km. An XDF single-mode ESCON channel is provided for connections up to 20 km. It is either a point-to-point connection or a system-to-ESCON Director connection.

Hardware: The ESCON Control Unit adapter (Type 5-3) FC #2756 is a Micro Channel adapter for RS/6000 systems.

Advantages: The ESCON channel provides a high-speed channel connection between RS/6000 servers and mainframes over long distances of up to 43 km. Like BMPX, ESCON provides CLIO/S features to replace slow TCP/IP applications, such as FTP, allowing data to be sent at a high speed on channel level.

Disadvantages: The ESCON Control Unit adapter is much more expensive than the BMPX channel adapter on the RS/6000.

Conclusion: ESCON might be used for mainframe-to-RS/6000 migration as well as for sharing data in a mainframe / RS/6000 environment where high performance and long distances are required.

4.5.11 References

AIX Versions 3.2 and 4 Performance Tuning Guide, SC23-2365

`http://www.rs6000.austin.ibm.com/hardware`

4.6 Graphics Adapters

IBM offers a broad range of versatile graphics accelerators to meet customers' application needs from entry 2D design and drafting to complex 3D solid modeling. Any member of the IBM RS/6000 family (except the rack systems, some SMP servers and SP systems) can include a graphics accelerator. When a graphics accelerator is included, the system is referred to as a workstation.

To perform the many calculations necessary for the creation of computer graphic images, graphics workstations use specialized graphics processors or adapters to offload graphics computation or processing from the CPU. These adapters may have their own memory subsystem and provide the graphics acceleration to produce graphics output (picture) at the graphics display monitor.

The graphics adapter is connected to the system CPU and system memory by a bus. IBM RS/6000 supports several types of bus attachments for graphics adapters: the Micro Channel bus, the 60X local bus and the PCI local bus. The Micro Channel provides a common method of attaching adapters and features across the RS/6000 family and makes it easy to maintain compatibility. The 60X local bus connects the graphics processor directly to the system memory and the system CPU. This allows for fast data transfer to and from the graphics processor with minimal system overhead. This connection method is the one used by some PowerPC RS/6000 machines. The PCI graphics adapters attach to the PCI bus on some PowerPC RS/6000 machines.

Graphics accelerators or adapters offer a wide range of performance and capability. From entry to high-performance 2D graphics accelerators, these adapters support the X Window System and provide acceleration for key 2D functions and operations. In addition, some of IBM's family of 2D graphics adapters also provide support for popular 3D APIs (OpenGL, PHIGS) when combined with optional IBM software (Softgraphics). The combination of an RS/6000 workstation and graphics adapter provides graphics functionality and performance as needed.

From entry to high-end 3D graphics accelerators, these adapters support open APIs such as OpenGL. These graphics accelerators are designed to offload the system processor by providing varying amounts of hardware acceleration.

From a 3D graphics perspective, IBM's 3D graphics adapters are classified into three basic hardware classes:

- Class I

  The adapters belonging to this class provide a frame buffer but no unique hardware for 3D graphics and perform all graphics operations (both geometry and raster processing) on the system CPU.

- Class II

  The adapters belonging to this class provide hardware rasterization for 3D graphics functions and rely on the system CPU for geometry processing.

- Class III

  The adapters belonging to this class perform all 3D graphics operations, both geometry and raster processing, on the graphics adapter.

Figure 39 illustrates the 3D graphics pipeline and maps of IBM's adapter classes.

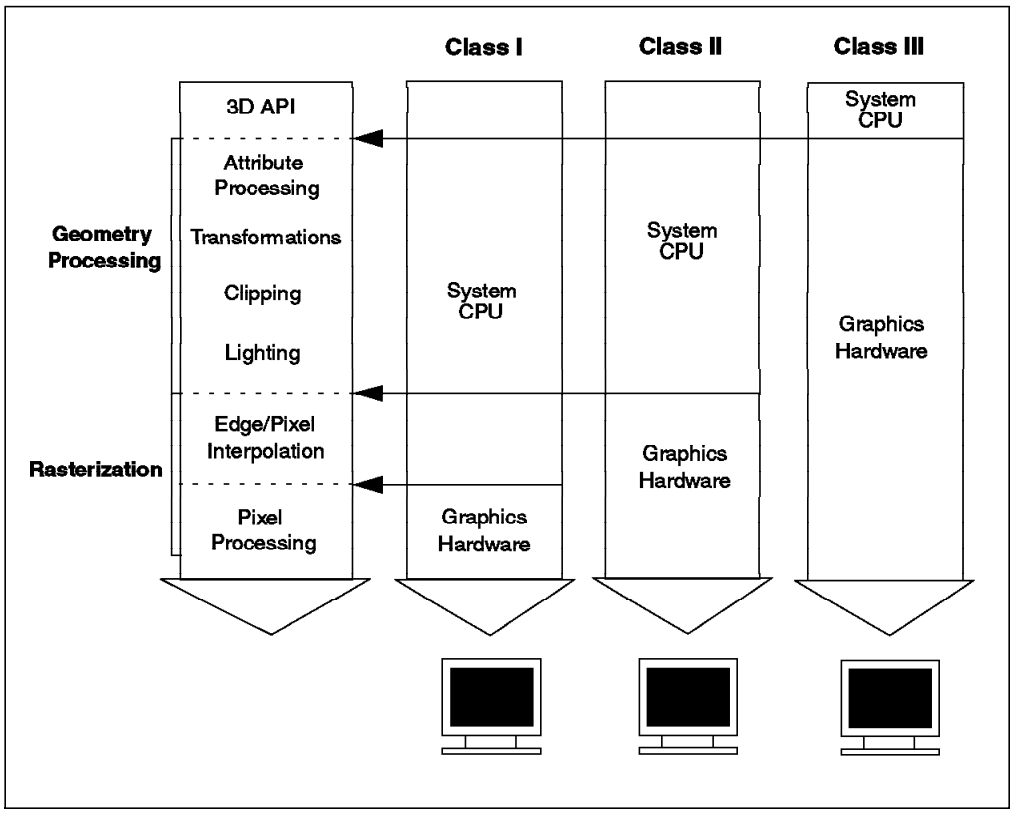


Figure 39. 3D Graphics Pipeline

This chapter will discuss the performance aspect according to the category or class of the graphics accelerators. This chapter will not discuss in detail the capabilities of the adapters (such as resolution, color support) and graphics applications. For detailed information (capabilities, features, and performance numbers) please refer to `http://www.rs6000.ibm.com/hardware/adapters/graphics` and its related links.

4.6.1 Entry Graphics Accelerators

The graphics accelerators belonging to this category are: E15 (integrated on the 43P-120 and 133), S15, and POWER GXT110P. All of these adapters except E15 require a single PCI bus slot. All these adapters are intended only to offer 2D graphics functionality and no 3D support (either hardware or software). These adapters provide good 2D performance numbers.

4.6.2 2D Graphics Accelerators and Entry 3D (Class I)

These adapters provide 2D graphics capability, and with the IBM Softgraphics software, all these adapters can run true 3D graphics applications. The RS/6000 provides two levels of 3D entry graphics products: POWER GXT15xx family and POWER GXT25xx family.

IBM's POWER GXT15xx family consists of POWER GXT100, POWER GXT150, POWER GXT150L, POWER GXT155L, POWER GXT150M, and POWER GXT150P. These adapters provide excellent 2D performance.

IBM's POWER GXT25xx family consists of the 8-bit POWER GXT250P and the 8-, 16-, or 24-bit POWER GXT255P. Both accelerators provide excellent 2D performance for the 43P-120, 133, 140, 240, and model F40.

All of these graphics accelerators support Softgraphics, an IBM software that provides 3D capability and supports OpenGL, PHIGS and PEX 3D APIs. Through this software, entry 3D graphics accelerators can provide good 3D performance without 3D graphics-specific hardware. It replaces the need for a 3D graphics accelerator by performing 3D graphics functions on the workstation CPU system and displaying this information via a 2D graphics adapter. Softgraphics delivers graphics performance that scales with the workstation processor and enables existing applications to easily exploit new adapters as they become available. The 3D API implementation of Softgraphics is portable with the 3D hardware graphics accelerators. This means the 3D graphics applications can be scaled according to the combination of CPU and graphics accelerators.

4.6.3 Mid-Range 3D Graphics Accelerators (Class II)

The IBM GXT family of 3D graphics accelerators for the RS/6000 that belong to Class II includes the POWER GXT500, GXT500D, GXT500P, GXT550P, GXT800P, and Gt4xi (24-bit). These graphics accelerators provide a higher level of 3D graphics performance. These adapters provide hardware acceleration support for antialiasing, texture mapping, and rasterization; in other words, they accelerate drawing operations in hardware (such as filling in the pixels of a triangle) while leaving geometry processing (such as lighting and transformations) to be done on the host CPU. This leads to a well-balanced system where graphics performance scales reasonably well with processor performance.

4.6.4 High-End 3D Graphics Accelerators (Class III)

At the high-end, Class III accelerators provide the highest level of graphics performance by providing hardware acceleration for both rasterization and geometry processing. These types of accelerators typically offload geometry processing onto several digital signal processors (DSP), which in turn make use of custom rasterization chips. Therefore, these accelerators tend to be insensitive to host CPU performance, and graphics performance is hardly affected by an increase in host CPU performance. By offloading graphics tasks, however, these accelerators free up the host CPU for non-graphics tasks. The POWER GXT1000 and the Freedom Series 6000 are representatives of this class of graphics acceleration. These adapters are very suitable for the most-demanding 3D graphics applications.

4.6.5 Graphics API Enhancement on SMP Systems

In October 1996, IBM announced the graphics API enhancement for SMP systems along with the announcement of a new SMP workstation. Currently, only the PHIGS API can utilize the multiprocessor feature to provide higher performance on an SMP workstation, and OpenGL enhancement is still in development.

Multiprocessing on a workstation machine has a different objective than on a server machine. As the workstation machine always uses the graphics adapter, whether it is integrated or through I/O bus (MCA, PCI), all the subsystems (graphics adapter, graphics software) contributing to the graphics should be designed to exploit multiprocessing capability on the SMP workstation. Typically, a workstation machine is used as a single-user machine with many windows displayed on the graphics display monitor. Therefore, the SMP workstation has a different workload from the SMP server and usually has fewer CPUs (two vs. eight).

The graphics library (PHIGS) can do multithreading on several CPUs in the SMP workstation. Figure 40 and Figure 41 on page 103 show the different run-time process structure of the PHIGS API on a uniprocessor system and on an SMP system.

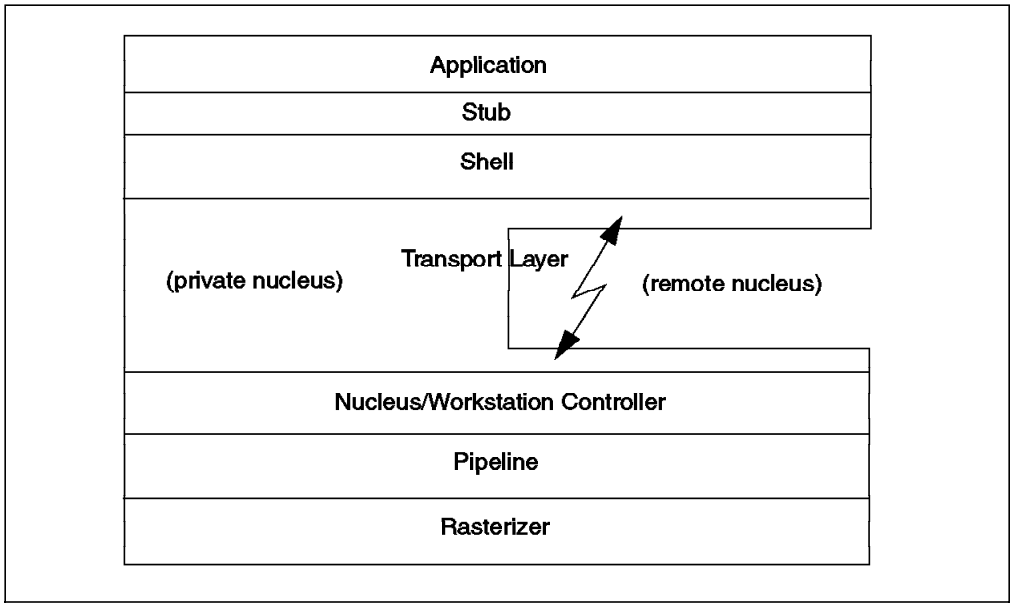


Figure 40. PHIGS API Run-Time Process on a Uniprocessor System

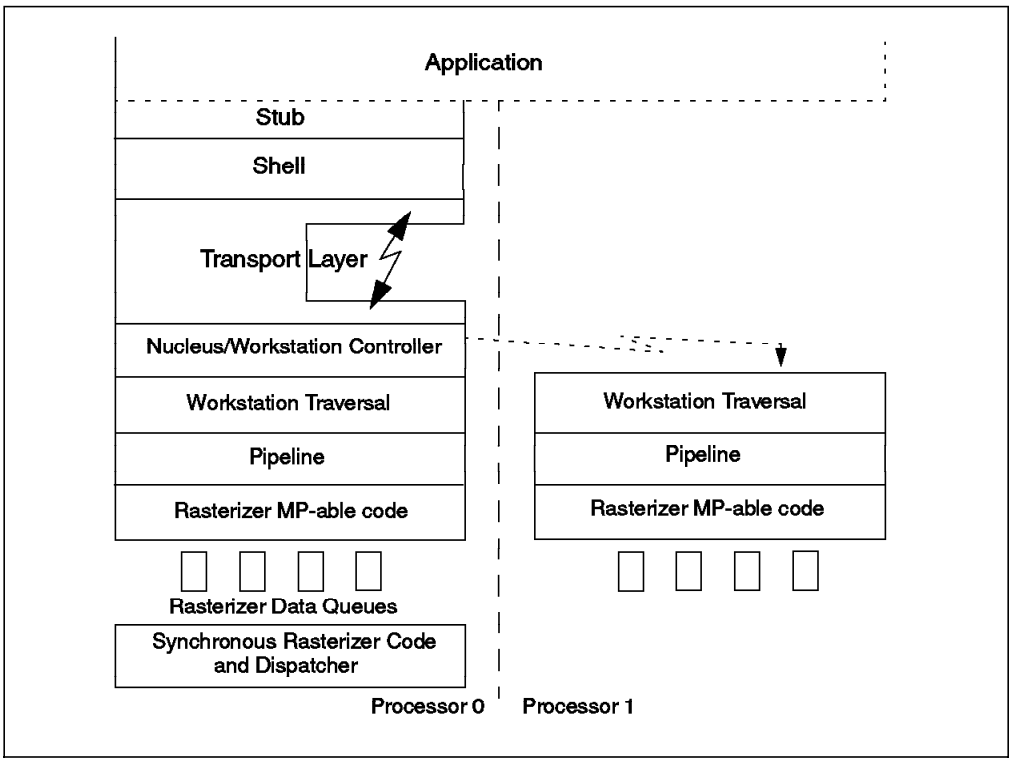


Figure 41. PHIGS API Run-Time Process on an SMP System

The issues and goals of the implementation are to maximize performance gains. The following issues of SMP systems have to be resolved:

- Avoid data movement between processors, since frequent data movement between L2 caches is expensive.
- Create threads once and keep them around because frequent creation and destruction of threads is expensive.
- Share resources of the machine with other processes (use dynamic scheduling).

Basically, the graphics API enhancement on the SMP workstation stems from multithreading of the pipeline and rasterization on several processors to gain performance. This implementation is applicable to Softgraphics (PHIGS, OpenGL), so the graphics applications will transparently gain the benefit of this feature. Applications do not have to be relinked to allow the PHIGS API to use *pthreads* (AIX thread library). If the application is relinked (on SMP machines), the PHIGS API will use multithreading within the application process. But if the application is not relinked (on SMP machines), the PHIGS API automatically forks to create a new process within which the PHIGS API will use multithreading. The PHIGS API will only utilize multithreading for GXT250, GXT500, GXT800 classes of adapters, as well as XSOFT and IMAGE workstation types, but not for GXT1000, because it is a separate graphics processor.

4.6.6 Understanding Graphics Performance

In order to understand graphics performance, it is necessary to first understand the environment in which the graphics hardware and/or software will operate. Factors which influence the user environment include: the application software, the system software (graphics API, programming libraries, windowing environments), the graphics hardware adapters, the workstation CPU and its configuration (memory, cache, and I/O bus).

The most-commonly used, and often misleading, method for describing graphics performance is hardware primitive rates. Hardware primitive rates are typically expressed as 2D vectors/second; 3D vectors/second or 3D polygons/second; and bit rates. Unfortunately, the characteristics of the vector and polygon rates differ from vendor to vendor. Some of these characteristics are:

- The graphics API used or whether programming has been done directly to the hardware.
- Whether display list rendering or immediate mode rendering was used.
- The length of vectors (in pixels).
- The type of vectors (2D, 3D).
- Vector orientation (horizontal, vertical, random).
- Portion of vector visible on screen.

When 3D primitives -- especially surfaces and solids -- are used, the list of variables increases. Standardizing vectors and polygons for measurement purposes would not solve the real problem.

If you do consider primitive rates, it is important that you define exactly what you compare. In general, we discourage the use of primitive rates but encourage the use of application benchmarks. Even a ″defined″ hardware primitive additionally depends on interactive performance, I/O performance and other factors. Also, hardware primitive rates are peak rates that can be measured in a laboratory environment but do not relate at all to the application's performance. Because of these reasons, IBM does not publish hardware primitive rates but instead supports the use of industry-standard benchmarks.

Besides the graphics performance, graphics or image quality is a factor that should be considered for a good graphics system. There are some cases where we can see insufficient image quality, such as:

- Poor hidden line and hidden surface removal.
- Insufficient antialiasing:
  - Line quality and color vary depending on the background color
  - Short points and lines ″jump″ when objects rotate
  - Lines become ″jaggies″
  - Line width appears to be a function of the line angle
  - Line roping
- Insufficient gamma correction:
  - Unnatural-looking line crossings
  - Dept cue objects get dim too quickly
  - Patterns show up where no patterns are defined

In fact, the quality of the original image is very important. High image quality can help to distinguish between rendering errors and errors in the underlying mathematical model. Unfortunately, graphics performance and image quality are a trade-off. We can make both excellent, but it will be very expensive, so a sensible compromise should be the goal.

4.6.7 Graphics Accelerator Hardware Sizing

In order to be able to do the graphics accelerator hardware sizing, we have to understand the user environment where the graphics system will be used. There are many combinations between workstation CPU type (PowerPC, POWER2) and graphics adapters (2D, 3D) we can choose. But often, they are interdependent because computer graphics performance must be approached as a complete system, not as the performance of an individual graphics adapter.

Usually, we can refer to the published graphics performance benchmark results to choose the graphics adapters. The following items are the factors that should be considered to size a complete graphics system.

- Application

  This is the first thing we have to know, and it is critical to know what applications the users are interested in using today and in the future. If actual applications are not specified, the users will have to describe what they want to do. The range of graphics applications are very wide, from 2D types to the most demanding 3D types.

- Graphics API

  Every graphics application requires a certain graphics API. Performance characteristics differ widely based on the API being used. If the user's application will be using X graphics calls, you may want to use XPC numbers, possibly in the form of a weighted average such as Xmark93 benchmark or, ideally, performance results from a similar application running on the platforms being considered. A good performance indicator for the PHIGS API is the PLB benchmark. For OpenGL performance, the OPC benchmark gives similar information. The different OPC viewsets have different characteristics. However, actual application performance comparisons are preferable.

- Graphics accelerator hardware

  Based on the information of the application requirements, it can be decided which type and class of graphics accelerator hardware will be needed. The performance of the graphics hardware depends greatly on what it is required to do. For instance, different APIs may require a graphics adapter to perform a similar task (such as drawing a line) in differing ways (usually for compatibility reasons). As a result, unique and often quite different performance results are obtained for similar tasks using different APIs. If the application only needs the 2D graphics capability, we can choose the 2D graphics accelerator that is suitable for the application. If the application needs 3D graphics capability, the next question is what class of graphics accelerator is needed.

  For most demanding 3D graphics applications, certainly, it will be appropriate to choose the Class III graphics accelerators. Certain 2D and entry 3D graphics adapters (Class I and II) can offer 3D graphics capability through software (Softgraphics), but this means the graphics performance depends on the main processor performance. There should be a compromise between expected performance and price consideration. Besides performance considerations, we have to understand some characteristics that might be needed by the application, such as texture mapping, graphics resolutions, color resolutions, graphics bit plane (8, 12, 24 bits), and buffering.

- Main processor and its subsystem

  In most cases, graphics performance is dependent on the system to which it is attached (the workstation). In this case, it is important to understand how much CPU power the application will require. This should be considered as well as the performance of appropriate benchmarks on varying CPUs when choosing a workstation.

  This is especially important if the graphics accelerator belongs to the Class I or Class II type, where the main processor performs some graphics

processing. It is not always true that all the graphics applications need high floating-point CPU performance. In cases where graphics applications will not need high floating-point performance, such as multimedia, graphical information systems, PowerPC workstations can be a good solution. If the application needs very high floating-point performance, then POWER2 workstations are suitable for this purpose. If there are other applications running on the same system, also consider the required CPU power and the amount of memory and disk space.

4.6.8 Conclusion

Graphics accelerators can be considered a subsystem that has its own processing (processor) to perform graphics functions. In order to get the highest possible graphics performance, it is very important to have a very good bandwidth (memory, I/O) between main CPU and graphics adapter. In fact, some IBM graphics adapters can perform DMA transfers to offload CPU transfer data from memory.

Graphics performance cannot be measured by hardware primitive rates because they are not a good indicator of application performance. Current industry-standard benchmarks are organized by the GPC group, and these benchmarks have several different types. The benchmarks are more application-oriented, but in some cases, they reveal only the performance of graphics hardware and graphics APIs and cannot be applied to real application environments. In order to get a more-objective graphics performance result, it is recommended to test the real application along with the exact workstation and graphics adapter. The overall application performance is determined not only by the graphics adapter but by the performance and capabilities of the operating system, CPU, I/O subsystem, bus, and memory. In other words, a good balance of the overall system can provide good graphics performance for an application.

4.6.9 References

`http://www.specbench.org/gpc`

`http://www.rs6000.ibm.com/hardware/adapters/graphics`

RS/6000 Technical Workstation and Graphics Marketing Guide (RS6KGRMG PACKAGE on MKTTOOLS)

Chapter 5. IBM RS/6000 Products

This chapter provides information on IBM SMP and IBM SP products and their performance-related features. An SMP-versus-SP selection guide also is provided in this chapter.

5.1 Symmetric Multiprocessor

The symmetric multiprocessor (SMP) RS/6000 servers are designed to spread a workload across multiple processors within a system, improving the overall performance. The RS/6000 SMP is based on PowerPC technology, initially on the 601 microprocessor and now on the 604 microprocessor. This provides an upgrade path for performance improvements for the installed base as well as giving scalability.

5.1.1 Hardware

This section examines memory-related performance issues and IBM's responses to those challenges.

5.1.1.1 SMP Cache Consistency Problem

The most basic problem that all SMPs must deal with is cache consistency.

Cache Coherency: Figure 42 on page 108 shows an example of the problem. Suppose process p1 is running on processor 1 and process p2 is running on processor 2. Suppose also that both processes p1 and p2 are working together on a problem sharing some memory. Consider the following sequence of events:

1. Process p1 loads address 123, which contains character "A".
2. Process p2 stores the character "B" into address 123.
3. Process p1 loads address 123 again.

The value seen by process p1 at step 3 is very important. With a naive implementation, p1 sees "A" because it has a copy of address 123 in its cache, the store request of p2 never goes out to memory, and p1 does not see the new value "B" that process p2 placed there.

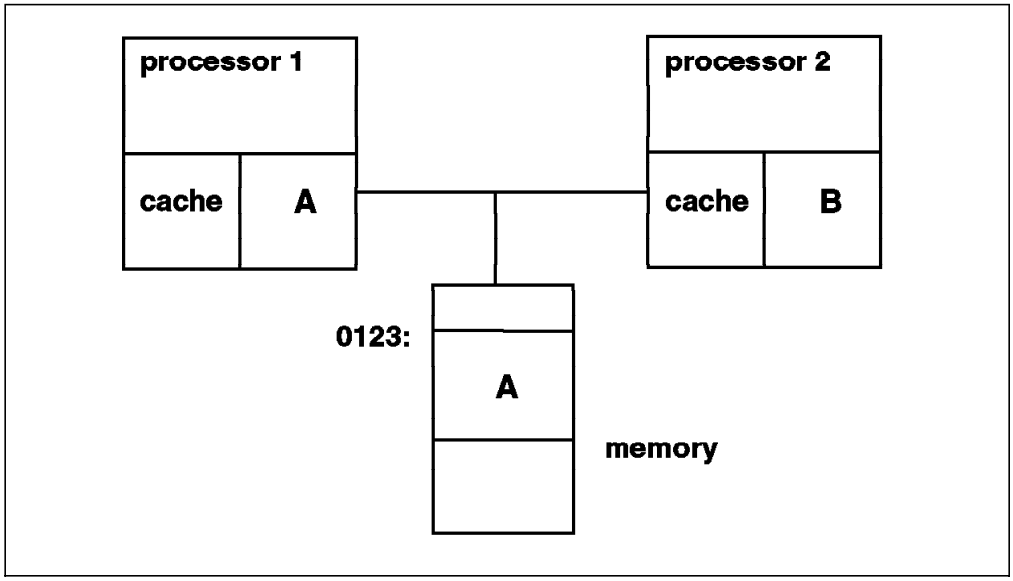


Figure 42. SMP Cache Coherency Problem

Snooping: One solution to the cache coherency problem is snooping. Snooping is a hardware logic at each processor/bus interface that broadcasts a message over the bus each time a word in its cache is changed. The logic also snoops on the bus for such messages from other processors. Whenever it detects that another processor has changed the value at an address that is copied in its own cache, the snooping logic invalidates that entry in its cache. This "cross invalidate" reminds the processor that the value in that location in the cache is invalid. If the processor needs to access this data, it will have to look for the value in another cache or in the main memory. Cache-to-cache data transfers are called ″intervention″.

In the example, when process p1 reads address 123 the second time, it gets a cache miss and must look elsewhere for the value. The extra snooping logic determines where process p1 should look to get the proper value for address 123. If the new value has not yet been written to memory, process p1 will obtain the value from the cache of processor 2.

Since cross invalidates increase cache misses and the snooping protocol adds traffic to the bus, solving the cache consistency problem reduces the performance and scalability of all SMPs. Therefore, IBM SMPs employ a specific technology, called the data crossbar switch, to alleviate the impact of cache consistency on performance.

False Sharing: The unit of access in the cache is called a line. A typical cache line on the RS/6000 machines is 32 bytes or eight words. It is possible for two processes to reference two different portions of data that fall in the same cache line because they lie close to each other in memory. For example, if a process on processor 1 changes the value of d1, the cache consistency logic will invalidate processor 2's cache line, causing a cache miss when d2 is accessed, even though the two processes were not sharing any data. This is called "false sharing". False sharing increases cache misses and bus traffic, further reducing SMP throughput and scaling.

The bigger the size of the cache line, the higher the miss rate. Some SMP implementations have a 256-byte cache line. The IBM SMP works with a 32-byte cache sector, and the coherency mechanism is based on the cache sector.

5.1.1.2 Memory Subsystem Performance

One of the techniques used to improve memory latency is to interleave the memory. This technique is not specific to the IBM SMP but is generally used by the industry. However, the IBM SMP implements a very high level of interleaving.

Let us suppose that the system has four 256 MB memory modules. Without any interleaving, each memory module would store a contiguous block of physical address space. In our example, the first module would store data for physical addresses *0x0* through *0x0FFFFFFF*; the second would store *0x10000000* through *0x1FFFFFFF,* and so on.

While this is simple, the main disadvantage is that accesses to adjacent addresses, which often happen within a short time due to spatial locality, will go to the same memory module. The memory module will be busy and will not be able to handle the request. Busy modules will increase the overall memory latency.

To overlap the memory cycle times better, memory is interleaved such that address space is striped across the modules. In this case, the amount of contiguous memory stored in a module is usually equal to the cache line size or the cache sector size. Since the cache sector is 32 bytes in our SMP implementation, the data for physical addresses *0x0* through *0x1F* would be stored in the first module, addresses *0x20* to *0x3F* in the second, addresses *0x40* to *0x5F* in the third and addresses *0x60* to *0x7F* in the fourth. Then, the addresses would wrap around to the first module again for addresses *0x80* to *0x9F*, and so on. The exact way in which the physical address space is interleaved among the modules is invisible to the software.

When interleaving is done with four modules, we say it is a 4-way interleaving. The IBM SMP system has a very high level of interleaving. According to the memory configuration, interleaving can be 1-way, 2-way, 4-way, 8-way or 16-way.

In summary, memory interleaving is a technique developed to allow simultaneous access to adjacent areas of memory. Interleaving also provides the ability to minimize “bank busy” effects in highly contended memories.

5.1.1.3 Data Crossbar Switch

A high number of cache “misses” is characteristic of commercial applications. Snooping activity, high intervention rates and transfers between memory and the I/O subsystem cause a high level of activity on the bus. If an SMP machine uses processors that are tied together using a bus, the memory bandwidth can become a potential bottleneck.

The IBM SMP is designed in a different way. There is still a bus for the snooping activity and the addressing, but a new component has been added for data transfers. That component is a switch called data crossbar (DCB) switch. The switch allows point-to-point connections between a processor and another processor or between a processor and the memory. It also allows several simultaneous transfers.

With such a technology, once the data is found, a point-to-point transfer can be done from the source to the requestor through the switch. While the switch is busy doing the data transfer, the bus is free for another processor request for data.

A switch has the following advantages:

- It removes work from the snoopy bus.
- It can transfer data among several units simultaneously.
- Connections are point-to-point, which allows a greater speed.

Figure 43, illustrates the use of the switch for data transfers.

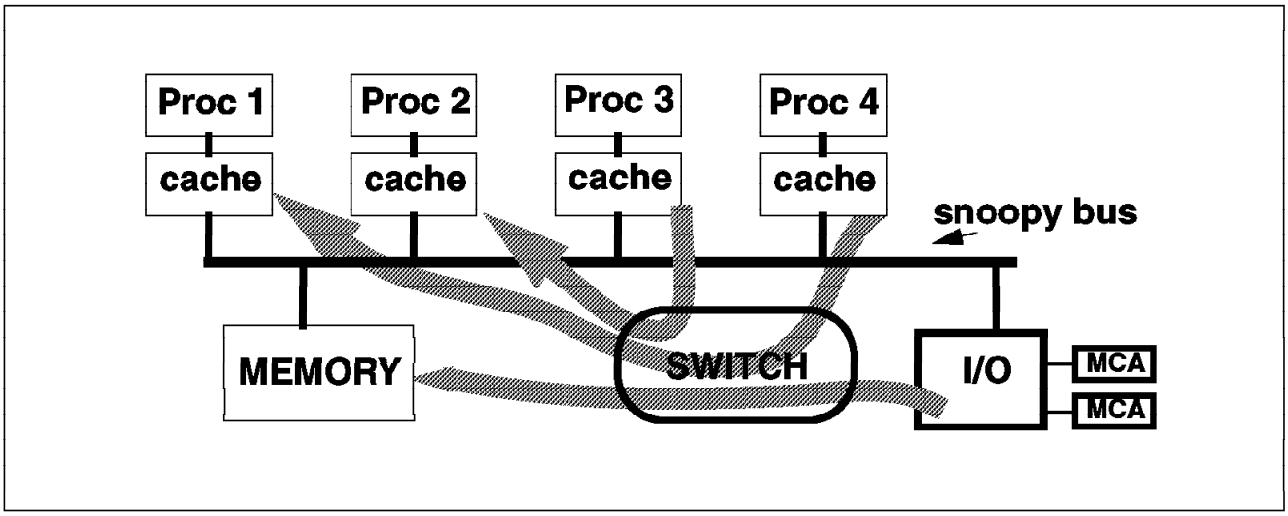


Figure 43. Using a Switch for Data Transfer

Crossbar Switch Performance: Crossbar switches do not eliminate data buses; they duplicate them to deliver more bandwidth. Every port on the crossbar is constrained to be point-to-point so that data transfers can operate at the bus speed (75 MHz). Four clock cycles are required per 32-byte transfer. This provides a peak transfer rate of 600 MB/s. Because each bank of memory has a bandwidth of 267 MB/s, to achieve this performance, the system must have at least four banks of memory (architecture requirement). Multiple crossbar ports provide exceptionally high peak system transfer rates.

The crossbar switch can handle two simultaneous memory accesses with a cache-to-cache (intervention) transfer. This results in a peak transfer rate of 1800 MB/s. Transfer rates above 800 MB/s require exploiting the larger cache line size of the 620. The SMP design architecture is able to evolve for faster processor clock and higher-performing PowerPC processors.

When transferring data through a bus, the latency depends on how much the bus is loaded. With a crossbar switch, the latency is fixed, meaning that loading data from memory takes a determined amount of time.

High bandwidth and fixed latency time give the IBM SMP a high scalability factor.

5.1.2 Software

Since most operating system activity is triggered by events, interrupts, and system calls, all processors are able to run any part of the kernel and access any kernel data simultaneously. To handle this activity correctly, changes must be made to UP operating systems that are going to be used with SMPs.

5.1.2.1 Threads and Locks

One of the main challenges is the implementation of threads. A thread is an independent flow of control within a process. All threads within a process can run concurrently on different processors. Threads are well-suited for exploiting SMP architectures. Since a classical UNIX process is considered as a single-threaded process, threads will be used in this section to illustrate some concepts.

SMP Synchronization Issue: There is a potential synchronization problem when two processors might try to update the same piece of data at the same time, and incorrect results can be obtained.

Consider an example where two threads are updating the same variable.

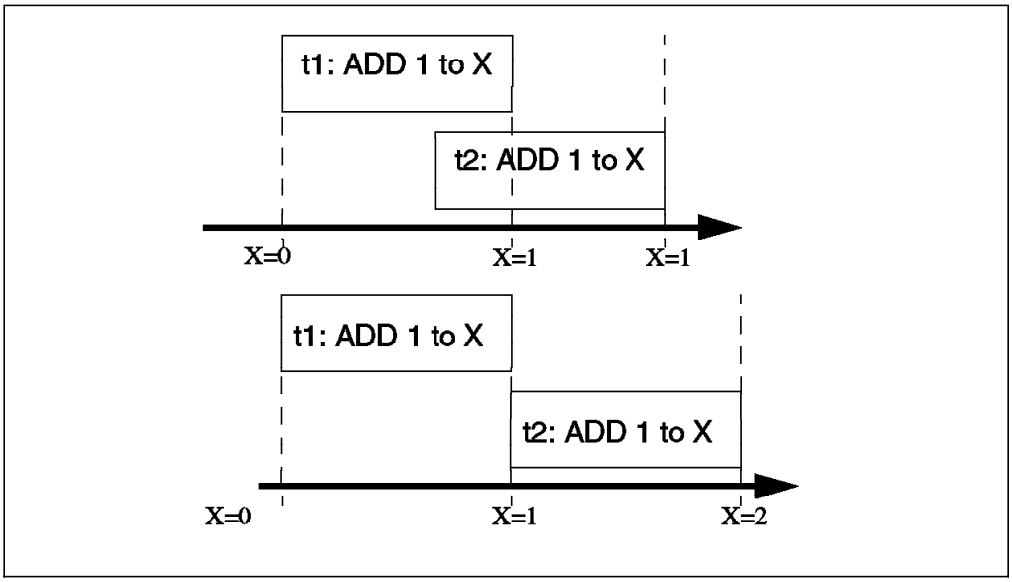


Figure 44. Synchronization Issue

In the top half of the diagram, threads t1 and t2 are both adding one to the same shared variable, X, whose value is n. The final value must be n+2, but t2 is incrementing the variable before t1 has finished. Therefore, the final value will be n+1. In order to avoid this, both threads have to be serialized, as shown in the bottom half of the diagram.

A critical section is a section of code that modifies shared data. Therefore, it must not be executed by more than one thread at a time. The other thread(s) must wait. In the above example, the critical section is the code that changes the variable X.

The problem of serializing access to shared data is generic to parallelized code. It occurs at both the user and the kernel level. This problem is resolved by locks on critical sections of code.

Conceptually, a lock is just a bit in memory that threads use to regulate their entry into critical sections. But locks are not that simple to implement because two or more threads could test the same lock simultaneously, determine that the lock is available, and enter the critical section. Because taking a lock requires several operations (read, test and set the lock bit), this operation is itself a critical section. Thus, multiprocessor hardware must provide a way to perform this test and set operation atomically with respect to the other processors. This

means that if more than one processor is trying to obtain the same lock simultaneously, exactly one of them will succeed.

The two major types of locks are the following:

- **mutually exclusive (simple) lock**

  It allows one process or thread at a time in a critical section.

- **read/write (complex) lock**

  It allows multiple readers into the critical section at once but guarantees mutual exclusions for writers.

Waiting for Locks: When a thread wants a lock that is already owned by another thread, the thread is blocked. It has to wait until the lock becomes free, and there are two ways of waiting: spinning or sleeping.

- **Spin locks** - These allow the waiting thread to keep its processor by repeatedly checking the lock bit in a tight loop (spin) until the lock becomes free. Spin locks are suitable for locks that are held only for very short times.
- **Sleeping locks** - The thread sleeps until the lock is freed, and it then is put back into the run queue. Sleeping locks are suitable for locks that may be held for longer periods.

Waiting for locks always decreases system performance. If a spin lock is used, the processor is busy but not doing useful work. If a sleeping lock is used, the overhead of context switching and dispatching, and the consequent increase in cache misses, will slow down performance.

Lock Penalty: Suppose that we know from tprof that when running a certain application, the system spends 10 percent of its time in a kernel component. Let us assume that the component is complex and touches a lot of data. The developer decides to make the whole component one big critical section. That is, there is only one mutex lock for the whole component, and it is requested at all entry points in the component and released at all exit points. On a 4-way SMP, this mutex lock will be busy 4 x 10 percent = 40 percent of the time.

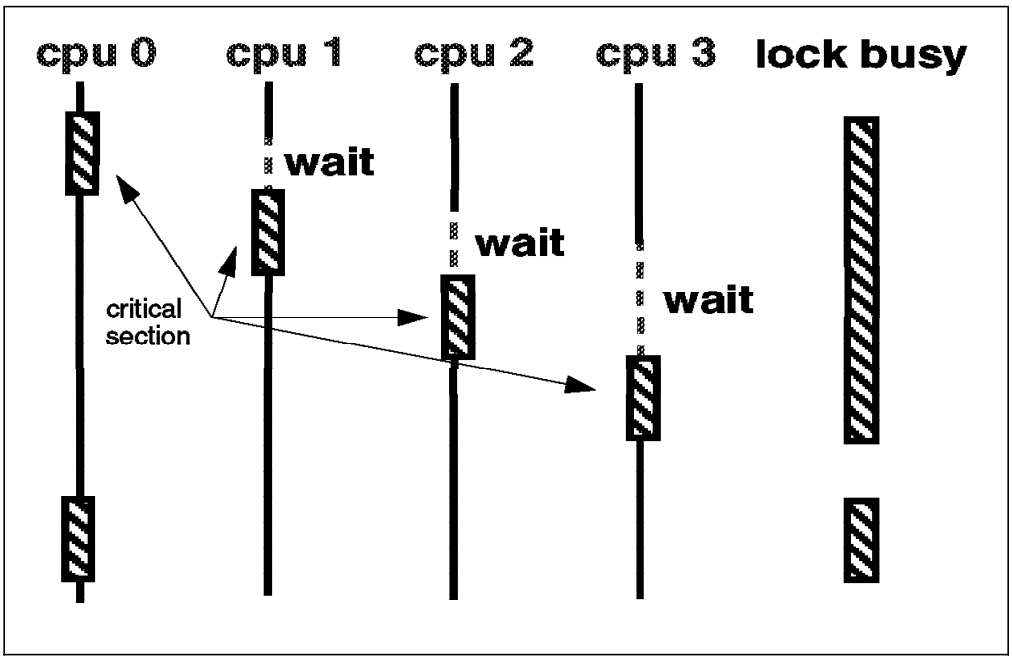


Figure 45. Lock Penalty

According to queuing theory: the busier a resource, the longer the average wait to get it. In addition, note that the relationship is nonlinear. If the use of the lock is doubled, the average wait time for that lock more than doubles.

The thundering herd problem occurs when several threads are queued waiting for a resource, the resource is freed, and then several waiting threads are awakened at the same time. For simple locks, this problem is avoided by selectively waking only the highest priority sleeping thread. For complex R/W locks, either the highest priority writer is awakened or all the readers are awakened if no writer is waiting.

Lock Granularity: The amount of time a lock is busy is a function of how often it is requested and how long it is held once acquired. The most effective way to reduce wait time for a lock is to reduce the size of what the lock is protecting. In other words, reducing the lock protection time reduces the waiting time.

Figure 46 illustrates lock granularity. Instead of locking the whole code routine, it is better to lock only the portions of code within the routine that actually modify shared data.

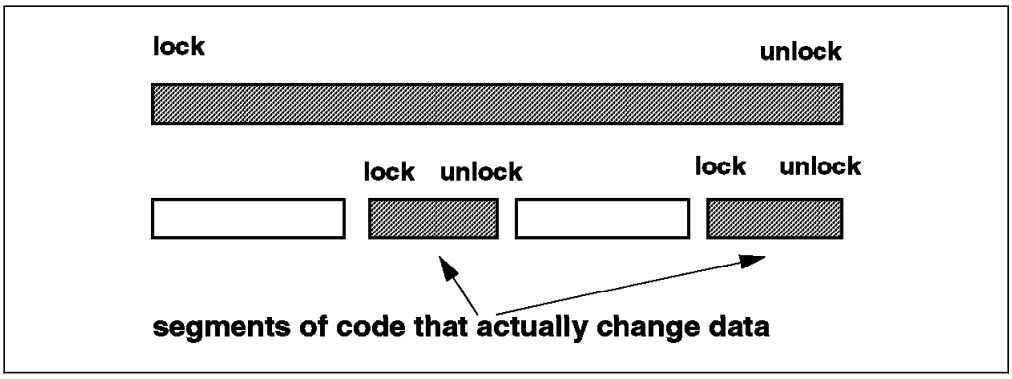


Figure 46. Lock Granularity

But, with granularity too fine, the frequency of lock requests and lock releases will increase. This adds additional instructions. Tests show that a lock/unlock pair costs approximately 20 to 125 instructions (added to the path length of the program). Therefore, if lock granularity is too fine, too many instructions are used to request and release the locks. If the lock granularity is too coarse, too much time is spent waiting for the locks. A balance must be found between a too-fine and a too-coarse granularity. This is again the developer′s responsibility.

Lock Performance Considerations: Here are a few performance tips for locks:

- Never perform synchronous I/O or any blocking activity while holding a lock.
- Move all the unnecessary instructions (those not directly related to reading or modifying the protected data) outside the critical section.
- If more than one access is needed to the same data in a given component, try to move the accesses together so they can be covered by one lock/unlock pair (provided there are not too many instructions).
- If more than one lock are to be held simultaneously, request the busiest one last, if possible.
- If the protected data is mostly read, consider using a complex lock instead of a simple one.
- The frequency with which any lock is requested should be reduced.
- Lock just the code that accesses the shared data, not all the code in a component (this will reduce the lock holding time).
- Locks should always be associated with specific data items or structures, not with routines.
- For large data structures, choose one lock for each element of the structure rather than one lock for the whole structure.

5.1.2.2 Processor Affinity

In AIX V4, the schedulable entity is the thread, and the thread with the highest priority is the one that gets dispatched. This means that a thread is bounced from one processor to another over its lifetime. As a result, it suffers many cache misses when reloading instructions and data on the processor where the thread is dispatched.

If we try to run the thread on the processor where it last ran, some of the instructions and data might still be in the processor cache. This technique may reduce the amount of cache misses and improve performance.

Affinity with a processor is the amount of data that is already in the processor cache. Processor affinity is the policy of trying to run a thread on the same processor where it last ran. AIX V4 has been changed to enforce affinity with the processors.

As shown in Figure 47 on page 115, run queues are ordered according to their priority, with 127 being the lowest and 0 being the highest. When a thread is dispatched from a queue on a processor, the identity of the processor is registered in the structure of the thread.

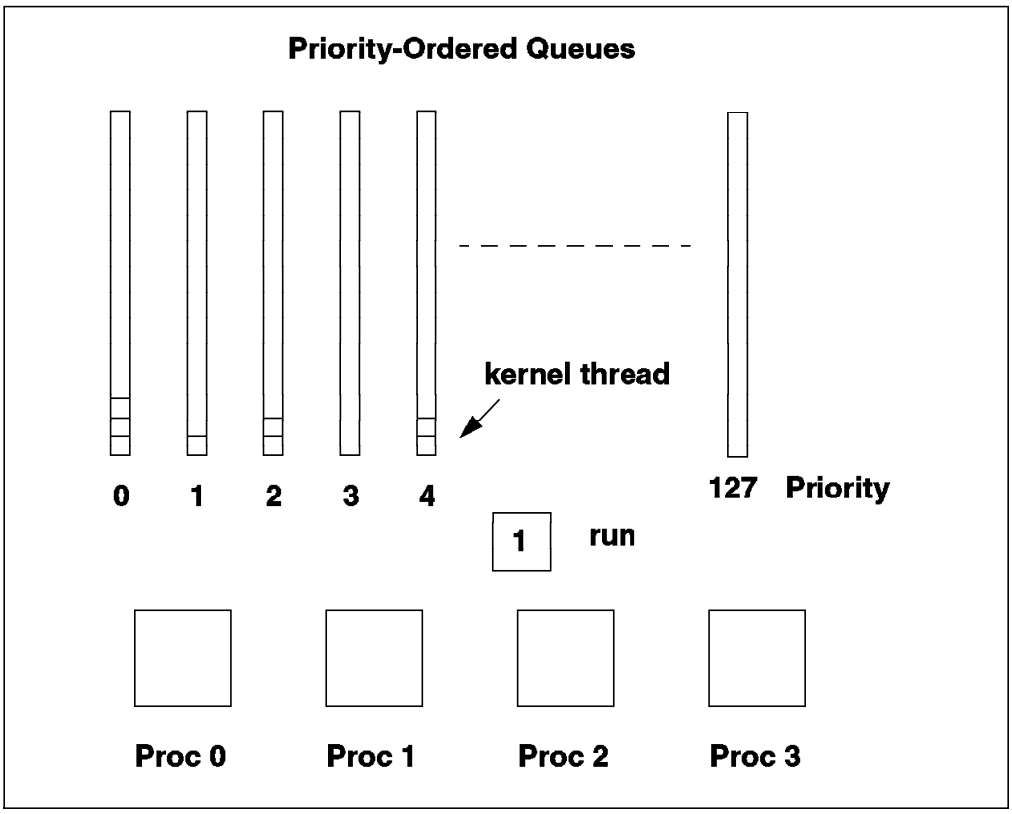


Figure 47. Threads Dispatching

In this way, each time the dispatcher selects a thread, it knows the processor number on which the thread last ran. When a processor asks to run a thread, the dispatcher chooses the thread with the highest priority from the priority-ordered run queues. It then tests to see if this thread has affinity with the processor.

If it has affinity, the thread is dispatched to the processor. If it does not, the dispatcher tries to find another thread which last ran on the processor by scanning the queues until it finds one. This scanning is not done indefinitely, and there are some limits:

1. If the priority difference between the thread with the highest priority and the thread that last ran on the processor is greater than a threshold value, the thread with the highest priority will be chosen. That threshold value is 0 by default. This means that the search for a thread with affinity with the processor is limited to the same queue.
2. Scanning is stopped when the number of scanned threads is higher than a predefined value. By default, that value is three times the number of processors (for example, 12 on a 4-way SMP).
3. Scanning is also stopped when the dispatcher encounters a boosted thread and the parameter `affinity_skipboosted` is `FALSE`.

   If a thread with a low priority holds a lock and if a higher priority thread is waiting for the same lock, the low priority thread gets the priority of the higher-priority thread (it is boosted so that the higher priority thread doesn't have to wait too long for the lock). This priority inversion is always done by the system. If the `affinity_skipboosted` parameter is set to `TRUE`, the boosted thread is skipped and the dispatcher goes to find a thread that has affinity

with the processor. To avoid running a lower-priority thread instead of a boosted thread, the default value of the parameter is FALSE.

5.1.2.3 Binding

Binding is the strongest form of processor affinity; it may be obtained by using the bindprocessor command or the *bindprocessor()* system call.

The bindprocessor command allows a user to bind all threads of a process to a specific processor. You cannot bind a process until the process is already running, so it must exist to be able to bind it.

Once a process is bound to a specific processor, it cannot run on another processor and take advantage of it. Binding might be useful for a process that seldom blocks for long periods and whose response time is important. However, binding a process may cause some performance problems by letting some of the processors remain idle. Binding is appropriate only in special circumstances, such as on a system that is dedicated to a single application.

The *bindprocessor()* call allows a developer to bind a thread to a specific processor at the programming level.

5.1.3 Scaling

One of the most important metrics of MP performance is scaling. When a processor is added to the system, how much additional performance is obtained on a given workload? Scaling is workload dependent. Some workloads will scale better than others. In addition, workloads will scale differently on different types of MPs. For example, a workload that shares a lot of data is likely to scale better on an SMP than on a shared nothing MP. This is because all processes on an SMP have a consistent view of the data, and processes on a shared nothing cluster must do message passing to share data when using a high degree of parallelism.

5.1.3.1 Scaling Myth

In a perfect world, one would expect SMP performance to increase linearly as processors are added. But this does not happen due to the overhead required to maintain a consistent view of the memory and other shared resources for each of the processors.

In an SMP configuration, programs share the operating system, memory subsystem, and disk I/O. Sharing means conflict, and that limits the number of processors that can be effective in a system.

The notion that SMP performance scales linearly is wishful thinking. Even the applications and benchmarks that show near linear performance must share the operating system and resources. Only those benchmarks or applications that spend very little time in the operating system, are cache resident, perform little I/O, have little main memory activity, and are CPU intensive may exhibit near linear scaling. Real programs use main memory, disk I/O and operating system services that would cause resource conflicts in an SMP system.

5.1.3.2 Scaling Limitation Factors

As more processors are added, each additional processor increases performance slightly less than the previously added processor. In fact, adding more processors ceases to boost performance after some critical number, as Figure 48 shows. In the worst case, a 16-way symmetric multiprocessor machine may provide less performance than a 12-way.

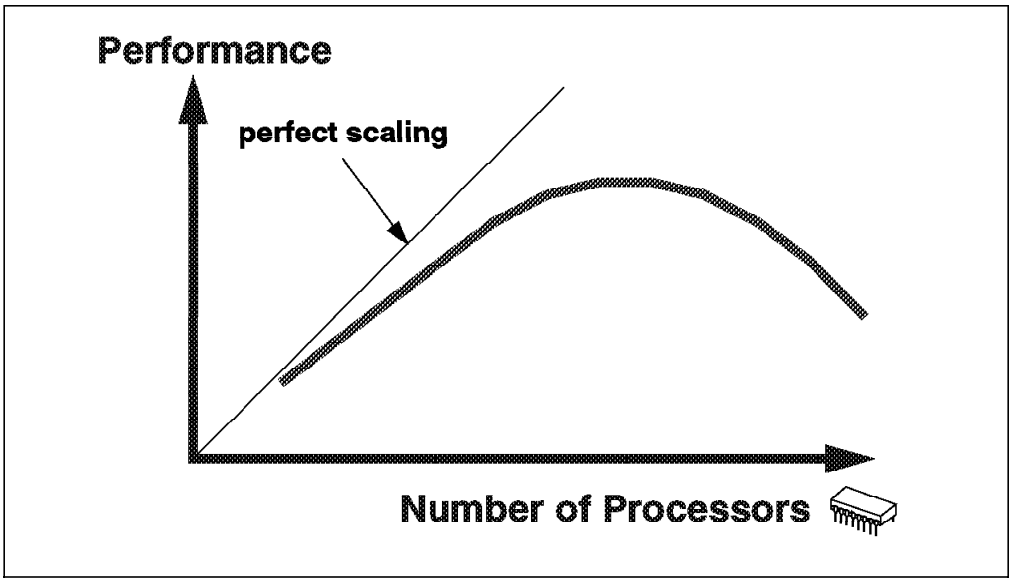


Figure 48. Scaling

There are many reasons why real workloads do not scale perfectly on an SMP system, and some of them are listed below:

- Increased bus/switch contention when the number of processors increases.
- Increased memory contention because all the memory is shared by all the processors. Memory conflict might occur when a processor needs to access a bank that is busy or some other memory component is locked. Obviously, a system that has many memory banks will scale better than a system that has one.
- Increased cache misses because of larger operating system and application data structures.
- Cache cross invalidates and lateral reads to maintain cache coherency. For example, if one processor requests a read of a cache line that happens to be modified and resident in another processor, the holder of the cache line will force the requestor to retry. The retry duration is not bound. Several retries may be needed to push a modified line down from a processor's L1 cache to the memory controller before being sourced to the requesting processor.
- Increased cache misses because of higher dispatching rates.
- Increased cost of synchronization instructions.
- Increased operating system and application path lengths for lock/unlock.
- Increased operating system and application path lengths waiting for locks.

It can be seen from some of the above factors that scaling is workload dependent. Some workloads may scale relatively well on an SMP while others will scale poorly.

5.1.3.3 Commercial vs. Technical Applications

Commercial applications have a number of characteristics. They use a large amount of data shared between many different users or programs. They have a low data locality, which means that there is a high level of data traffic between system memory and CPU caches, and there is a high level of I/O activity. There is also a high level of data traffic between caches due to lateral process migration. Therefore, a commercial application needs big L2 caches and very high bandwidth between memory and CPU as well as between the CPUs themselves.

Technical applications are usually CPU bound, and so processor speed is the key. Code is often made with short loops of instructions and may fit in the L1 cache.

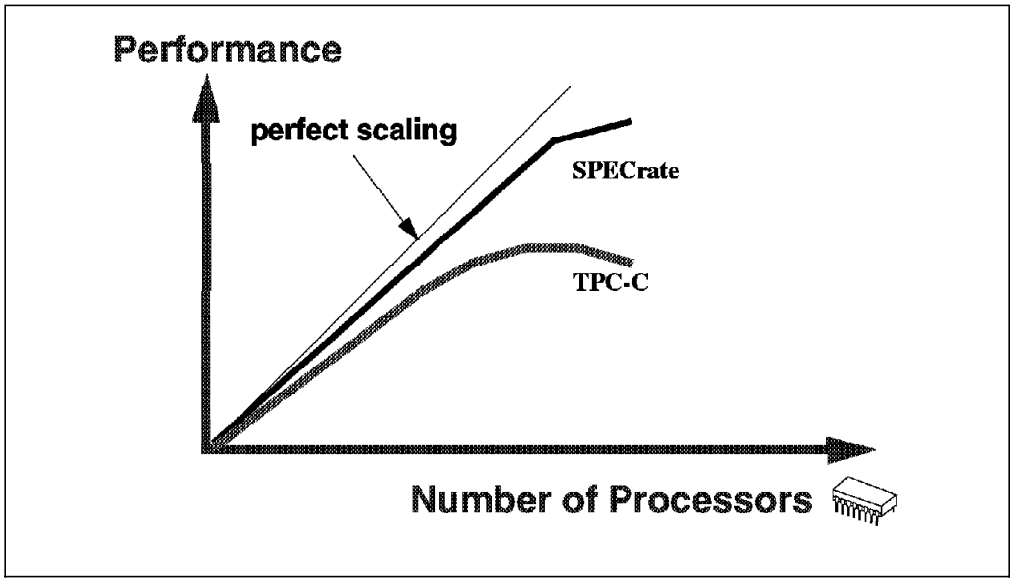


Figure 49. Scaling is Workload Dependent

5.1.3.4 Workload Scaling Example

The five workloads in this example are SPEC95, SDM benchmark, Order Entry Transaction Processing (OETP) and SFS.

SPECint_rate95 and SPECfp_rate95 execute overwhelmingly user-mode code, thereby putting little stress on the operating system. The SDM benchmark includes 057.sdet and 061.kenbus1. This group uses the operating system considerably. Approximately 50 percent of the execution time is in the operating system, stressing the virtual memory manager, file system, disk I/O, and dispatching. The OETP benchmark is unique in that it does considerable disk I/O but does not use the virtual memory manager or file system. This is because the database implements many functions normally done by the operating system. The OETP benchmark is a database workload defined with multiple transaction types with a scalable database. This benchmark executes about 15 percent of its time in the AIX kernel, primarily doing disk I/O and network communications. The SFS benchmark executes 100 percent of its time in the operating system. This is because NFS is implemented as a kernel extension within the AIX kernel address space. SFS stresses many parts of the operating system, including the virtual memory manager, file system, processor management, disk I/O, and network subsystems.

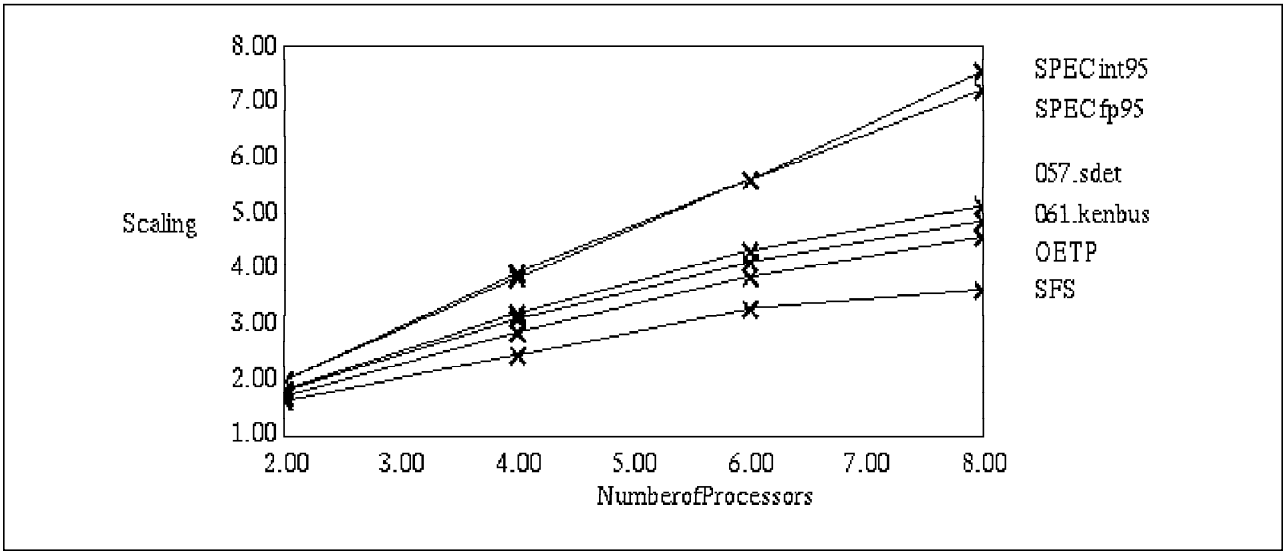


Figure 50. Workload Scaling

Looking at Figure 50, we can put workloads into three categories: high scaling, including SPECint_rate95, SPECfp_rate95; middle scaling, including 057.set, 061.kenbus, and OETP; and low scaling, which is SPEC SFS.

5.1.3.5 Scaling Metric

There is no universally accepted metric for scaling. In general, a one-way SMP will run slower (about 10 to 15 percent) than an equivalent processor running a UP version of the operating system. This happens because of the MP overhead that is inherent in the kernel of the MP operating system. So, as a result, most vendors will show scaling starting from two processors.

The following table is a hypothetical representation of how scaling can be represented. Having a ratio of 6.62 for eight processors shows that the OLTP benchmark scaling is good on the RS/6000 J40 machine.

Number of Processors	Relative OLTP Performance Value	Ratio to 1 Processor	Ratio to Number of Processors
2	5.8	N/A	N/A
4	10.0	3.44	0.86
6	14.5	5.00	0.83
8	19.2	6.62	0.82

Table 12. SMP OLTP Scaling Metrics For J40

5.1.3.6 Two-Dimensional Scaling

Most vendors can scale in one direction only, by adding more processors.

The IBM RS/6000 SMP servers allow two-dimensional scaling by being able to utilize higher-performance processors as well as by increasing the number of processors that can be added. These SMPs have been designed to allow for three generations of PowerPC chips to be included in the system (601, 604 and 620) and to support up to 16 PowerPC processors. The memory subsystem has

been designed to cater to that growth. Here is an example of faster processor scaling for the J40.

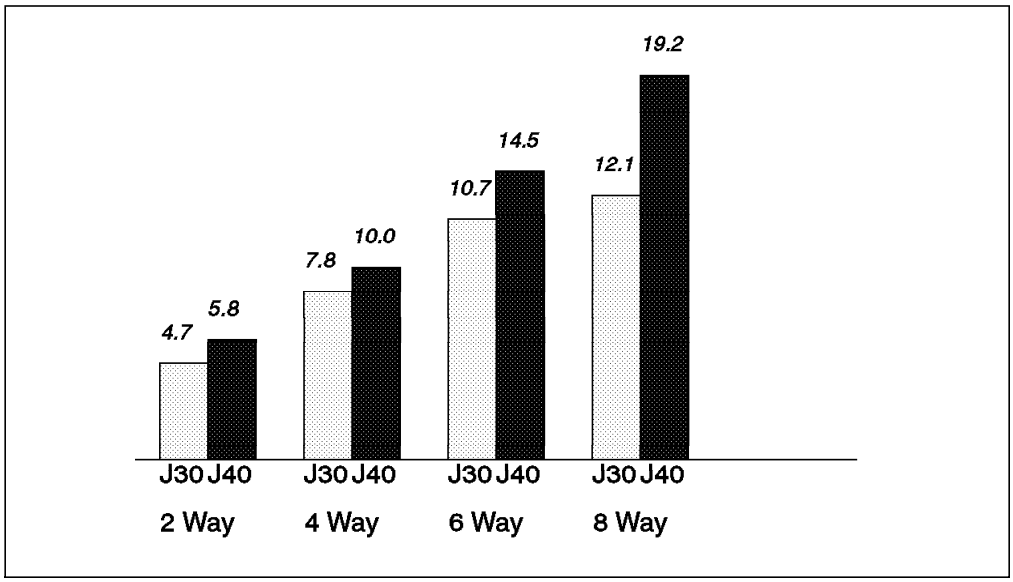


Figure 51. Relative OLTP Performance J40 vs. J30

5.2 Using an SMP

In order to effectively use an SMP, there are a number of things that need to be considered, such as parallelizing the application and Amdahl's Law. These are discussed in more detail in the following sections.

5.2.1 Parallelizing an Application

Parallelizing an application is one opportunity to effectively use an SMP. There are two ways to achieve this:

1. The traditional way is to break the application into multiple processes. These processes communicate using pipes, semaphores or shared memory. The processes must be able to block events, such as messages, from other processes, and they must coordinate access to shared objects with something such as locks.
2. The other way is to use the POSIX threads. Threads have coordination problems similar to those in processes and similar mechanisms to deal with these problems. Thus, a single process can have any number of its threads running simultaneously on different processors. Coordinating them and serializing access to shared data is the developer's responsibility.

Threads and processes each have advantages and disadvantages to be considered when determining which method to use for parallelizing an application. Threads may be faster than processes, and memory sharing is easier, but a process implementation will distribute more easily to multiple machines or clusters. Most RDBMS vendors use an implementation of "multithreading" which is their own design of threading and not true kernel threading. This allows an easier port to several platforms. But it is likely that, in the future, standard POSIX threading will be used.

5.2.2 Amdahl's Law

Amdahl's Law quantifies the fact that if only a part of the program speed is increased, the part that was not increased still runs as slowly as before.

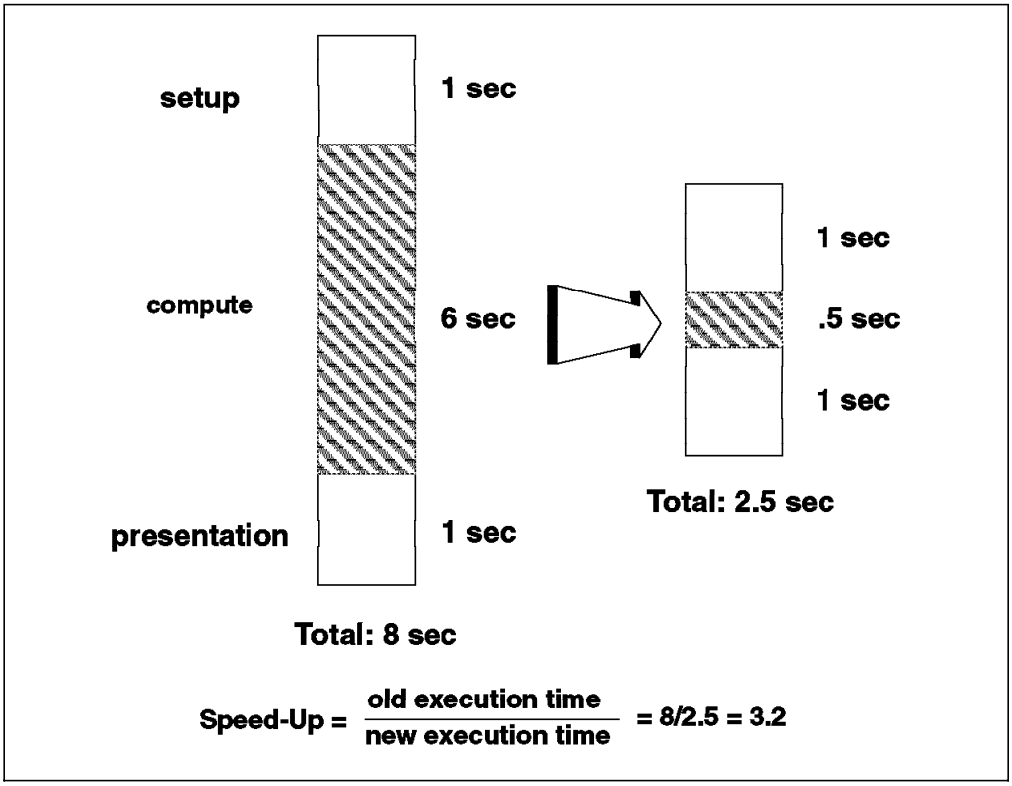


Figure 52. Amdahl's Law

In the above diagram, making the main part of the program run 12 times faster sounds good, but it only makes the program run 3.2 times faster. There is an upper limit on the increased speed that can be achieved by parallelizing the compute phase of this application.

5.2.3 SMP and Database

Although SMP originated in scientific computing, its true advantages are in transaction processing and databases.

The key variables for optimizing database scalability in an SMP system are techniques for deferring or avoiding I/O, reducing memory contention, reducing locking overhead and performing work in parallel.

5.2.3.1 Enhancing Performance

As we have seen before, fine-grain locking strategy can limit the amount of memory and the length time that memory is locked, thereby reducing its potential contention with other transactions. Three techniques to increase locking granularity are described below.

Multiple Shared Memory Locks: Each data structure in shared memory can be controlled by its own lock rather than by a common lock shared with other data structures. Therefore, a process attempting to update the lock table to lock a record does not need to wait for another process to finish updating the transaction table. Both processes can proceed in parallel. The resulting

decrease in contention for shared memory can have a significant impact on throughput and response time, particularly on SMP machines with many users accessing the databases concurrently.

Granular Transactions: Only the parts of a transaction that write data require exclusive access to in-memory resources. By locking the data for only part of the transaction (that is, by making the transaction more granular), the lock time can be dramatically reduced, and more transactions can use the same collection of in-memory structures at the same time.

Record Level Locking: RDBMSs that lock at the page or block level encounter serious contention problems when a transaction touches more than a few seconds. Since they must lock an entire block, all records in the block are locked for the duration of the transaction. As more transactions begin, the frequency of contention increases exponentially, resulting in a massive gridlock of transactions waiting for needed records. This can devastate response time and throughput. If a transaction only locks a particular record, the frequency of contention is dramatically reduced.

5.2.3.2 Multithreaded Operation

A multithreaded operation enables a server to accept multiple instruction streams from several clients simultaneously. The technique reduces CPU idle time (such as when a CPU waits for a disk I/O to complete) and serves multiple users concurrently.

Single-process multithreaded architecture reduces system overhead because thread switches can be done without entering the operating system kernel. Threads also have a much smaller memory context than processes. Thus, a thread context switch time is a small fraction of a process context switch time. This is advantageous because RDBMSs spend much of their time context switching. Time savings represent only part of the advantage of multithreading for a database. More significantly, the server can schedule threads according to the best use of the server. For example, suppose 1000 clients are each supported by one thread in the server. If one thread obtains an exclusive lock on a critical centralized resource such as the log buffer, a server process can guarantee that the thread will not be context switched while holding the critical resource.

This guarantee cannot be made in an RDBMS, which implements this using 1000 server processes. From time to time, the operating system will time switch a process holding a critical lock. The process allowed to run in its place will simply note that it cannot obtain the lock and give up the processor to the next process, which does the same, and so on. As the number of user sessions increases, the problem becomes worse.

Large Buffer Pool: The larger the memory buffer available to the RDBMS, the more of a database that can be kept in memory. With more memory available, the probability is reduced that a particular part of the memory will be required by two processes at the same time, thereby reducing the frequency of contention.

5.3 Parallel Architecture

The speed of conventional computers has increased tremendously over the years, but they are still considered not fast enough to reach the level of performance required to solve some complex computations or run highly computation-intensive environments. As we have seen, there are several ways to increase the speed of computers. Increasing the speed further would be difficult and very expensive because of the limitations inherent in the architecture upon which conventional computers are built.

5.3.1 IBM Scalable POWERparallel (SP) System

To overcome the limitation described above, the parallel architecture was introduced and implemented in several ways. The IBM SP (Scalable POWERparallel) system is one of the parallel architectures implemented in the IBM RS/6000 family.

The SP architecture is considered a distributed memory computer. In this architecture, all processors have their own memory in addition to other resources, such as I/O devices. They are commonly referred to as ″nodes″ of the parallel machine and are, in fact, independent computers sharing nothing else except a network. The network itself can be something from a LAN to an Omega switch network, which is the SP switch. Data is shared between processors through a message-passing type of communication protocol. Since almost nothing is shared and the network is only occasionally used compared to the traffic on a bus, the SP system is highly scalable. The simplified schematic of the SP system can be seen in Figure 6 on page 17.

While the SP is designed to provide a parallel environment, it is also effective for serial workloads and for both batch and interactive applications. The system has an architecture with significant growth capabilities and is based upon proven RS/6000 technology and AIX software. It also features innovative, topology-independent switches for high-speed, interprocessor communication for parallel computing, and a sophisticated set of IBM-developed software tools for system management, job management, and parallel application development and execution. Because the SP is also an open system, popular parallel interfaces from other sources are supported.

The basic SP building block is the processor node. It consists of a POWER2 microprocessor or PowerPC symmetric multiprocessor (SMP), memory, disk, and Micro Channel expansion slots for I/O, connectivity, and the SP switch adapter. Node types may be mixed in a system and are housed in short or tall system frames. Depending on the type of node selected, an SP frame can contain up to 16 nodes. These frames can be interconnected to form a system with up to 128 nodes (512 by special request). Currently, a maximum of 16 SMP high nodes can be installed per system.

A frame is vertically divided into drawers that span the internal width of the frame. The bottom-most drawer is smaller than the remainder and will accept a switch unit. The remaining drawers, if occupied, either contain one wide node or two thin nodes. A thin node is half the width of a wide node. A high node occupies four thin-node drawers or two wide-node drawers. All node types can coexist in a single SP system with the exception that, currently, the high node is not available for the short frame. The number of nodes of any type can be increased incrementally to meet computing power requirements for interactive,

batch, serial and parallel jobs, which can all be run simultaneously. Please refer to the RS/6000 Configurator Program for the node restrictions.

All SP node types are equipped with Ethernet adapters. Higher network data transfer rates may be achieved using other standard network types, such as token ring, fiber distributed data interface (FDDI), asynchronous transfer mode (ATM), or high performance parallel interface (HIPPI), which require adapters that occupy one or more Micro Channel slots. Substantially higher internode communication performance, a prerequisite for true parallelism in most applications, is obtained using the IBM SP switch. Each node gains access to this high-speed network through an adapter that occupies a single Micro Channel slot. The availability of the SP switch unit has been enhanced by improving circuit reliability, increasing redundancy, and reducing the global impact of error detection.

An important characteristic of the SP switch is that its bi-directional bandwidth is designed to scale linearly to thousands of nodes, with an essentially constant latency per connection. Hence, the necessary balance between interprocessor communication speed and the total system computation power is retained as the number of processor nodes is increased. This is why the SP is a truly scalable parallel system.

An RS/6000 workstation can be used to control and monitor an SP system. Typically, one control workstation can be assigned to serve 64 nodes. Central control is provided by the IBM Parallel System Support Programs (PSSP) executing on one or more control workstations. The control workstation connects to the frame through an RS-232 link, and one separate RS-232 link is required for each SP frame. So, if four frames are used, the control workstation should be configured with at least four tty ports.

A wide range of storage devices can be attached to the SP system through the SP nodes. The storage devices that can be attached are: the 7133 serial storage architecture (SSA), the 7134 SCSI storage subsystem, the 7135 RAIDiant Array, the 3995 optical library dataserver and the 3494 tape library dataserver. Up to 34 terabytes of disk storage can be attached to the SP system.

5.3.2 SP Switch Performance

The SP switch provides a high-performance interconnection between the processors of the SP. This is essential if acceptable parallel application performance is to be obtained. In parallel systems, it is usual to discuss the performance of an interprocessor communication fabric in terms of latency and bandwidth. Latency is the overhead associated with sending data between two processors, typically measured in microseconds (10-6 s). Bandwidth is the rate at which data can be transmitted between two processors, typically measured in MB/s.

Latency and bandwidth over the SP switch will vary depending on how it is measured. At the lowest level, we can consider the latency and bandwidth of the switch hardware. At higher levels, which include the SP software, users may access the SP switch via several different software communication paths. These paths lead to different communication performance.

In this section, we will describe the communication performance of the SP switch for the key communication protocols available in the SP. These include the fastest protocol available, called user space, and TCP/IP performance. The

high-performance user space protocol is most commonly used for technical and scientific applications, via a message-passing interface. TCP/IP is an industry-standard protocol and forms the basis of many applications, such as NFS and FTP, which are used to support file systems and file transfers. TCP/IP is also widely used in parallel commercial database applications.

Performance data for these protocols is presented in Table 14 on page 126 to Table 16 on page 127 below. Measured data were obtained on AIX Version 4.1.4 systems running Version 2 Release 1 of the Parallel Environment for AIX with PTF 12 in May 1996.

Hardware: Peak Communication Performance: The SP switch is a multistage network that is designed to allow communication bandwidths to scale linearly to thousands of processors. Scalability is achieved by increasing the number of switch stages as the number of processors increases. For each additional switch stage, there is an additional latency of 267 ns. This is not detectable at the application level. As shown in Table 13, the hardware unidirectional bandwidth is constant at 150 MB/s; the bi-directional bandwidth is 300 MB/s. This can be considered peak performance, and it usually cannot be obtained by an user application.

Number of Nodes	Latency (microseconds)	Bandwidth (MB/s)	
		Uni-directional	Bi-directional
2 - 80	1.2	150	300
81 - 512	2.0	150	300

Table 13. Hardware Latency and Bandwidth of the SP Switch

Several factors contribute to the communication performance that can be obtained by a user application. First, the switch is not the only piece of hardware to be considered. Communication performance depends on the processor, the memory subsystem, and the switch adapter. Therefore, when considering communication performance measurements, it is extremely important to understand the exact configuration of the system to which the data applies. In addition to hardware considerations, the system software contributes to the overhead involved in sending data between processors.

MPI/User Space: The Fast Path Over the Switch: On a distributed memory system like the SP, parallel scientific applications perform interprocessor communication via some form of message passing. Several different message-passing protocols exist. AIX PVMe, IBM's implementation of Parallel Virtual Machine (PVM) and Message Passing Library (MPL), are two examples of message-passing interfaces that can access the SP switch through user space. Recently, Message Passing Interface (MPI) has been adopted as the industry standard for message passing, and it is expected that, over time, applications using MPI will predominate. IBM fully supports MPI on the SP via the Parallel Environment for AIX software product, so we will discuss the performance of the SP switch for scientific applications in terms of what can be measured using MPI. Note that interprocessor communication performance measured using MPL is very close to the performance measured using MPI.

Table 14 on page 126 shows interprocessor communication performance measured from a Fortran program with MPI calls, using user space. Latency is

measured by sending a 0-byte message between two processors using *mpi_send* and *mpi_recv* from the MPI library. It is calculated as half the time for a round trip between the processors for that 0-byte message. Latency represents the time taken to set up a single message for transfer at the level of an application, and it may be seen as the overhead involved in transferring information between processors.

SP Processor Type	Latency (microseconds)	Bandwidth (MB/s)	
		Uni-directional	Bi-directional
77 MHz Wide Node	36.7	102.9	112.9
66 MHz Thin Node2	40.7	80.5	86.5

Table 14. MPI/User Space Switch Performance

It can be seen that the latency measured using MPI over the user space interface depends on the processor type. For example, the latency of the 66 MHz Thin Node2 is 41 microseconds, while the latency of the 77 MHz Wide Node is only 37 microseconds. This difference is largely due to the fact that the communication software is more closely tuned to the processor structure of the wide nodes. The difference between the thin and the wide processors is due to the increased speed of the 77 MHz processor. This dependency of processor type can be seen in all latency data presented in the following tables.

Table 14 also contains data for bandwidth measurements using MPI over the user space interface. These bandwidths represent the asymptotic data transfer rate that can be attained from a user application. The measurements were made from a FORTRAN program, with calls to the MPI library. The uni-directional bandwidth, sometimes called point-to-point bandwidth, was measured between two processors using *mpi_send* and *mpi_recv* calls, for message sizes of several MB. The bi-directional bandwidth, sometimes called exchange bandwidth, was measured in a similar fashion but with *mpi_isend* and *mpi_irecv* calls. These calls are designed to allow asynchronous communication, which means that data can be simultaneously sent and received by a processor. Thus, bi-directional bandwidth is the rate at which data can be exchanged between two paired processors.

It can be seen that there is a slight difference in uni-directional and bi-directional bandwidths. This is largely due to the fact that the communication software can handle multiple messages more efficiently than a single message. A more important factor is the correlation between bandwidth and processor type. For bi-directional bandwidth, the range is between 86.5 MB/s for the 66 MHz Thin Node2, and 112.9 MB/s for the 77 MHz Wide Node processor. The interprocessor bandwidths that the thin nodes can sustain are limited by their memory bandwidth. The transfer of data from the processor, to the adapter, onto the switch is very dependent on the rate that the processor can perform memory copies, hence the strong dependency on memory bandwidth. The memory bandwidth of processors in Table 14 are as follows: 66 MHz Thin Node2, 1.1 GB/s; 77 MHz Wide Node, 2.5 GB/s. For the wide nodes, the sustainable bandwidth is not limited by their processor memory bandwidth but by the Micro Channel, which is now being run at 160 MB/s.

MPI/UDP: Multiple Paths Over the Switch: The user space interface to the switch is optimized to provide a high-performance message-passing path for a single user application. To allow multiple users to access the switch from the same processor simultaneously, another path is available through UDP. Not only does this provide multiuser access, it allows other message-passing protocols that use IP, for example, public domain PVM, to run without change on the SP switch. This path is not intended to provide the same level of high-performance access to the switch as user space.

In Table 15, the interprocessor communication performance that can be obtained using MPI to access the UDP interface to the SP switch is shown. This data follows all the same trends described above for the user space data. The same dependencies on processor memory bandwidth are clearly seen.

SP Processor Type	Latency (microseconds)	Bandwidth (MB/s)	
		Uni-directional	Bi-directional
77 MHz Wide Node	255	18.8	28.8
66 MHz Thin Node2	293	15.3	23.0

Table 15. MPI/UDP Switch Performance

TCP/IP Performance: The SP supports the full set of IP protocols over the SP switch. Table 16 summarizes the latency and bandwidth of the TCP/IP socket protocol over the switch for this new release of the SP systems. These have been measured with the NetPerf benchmark (a public-domain benchmark). In this table, latency is half the round trip time for a 1-byte message, and bandwidth is the maximum obtainable both uni-directional and bi-directional over a single TCP socket (or stream) between two SP nodes.

SP Processor Type	Latency (microseconds)	Bandwidth (MB/s)	
		Uni-directional	Bi-directional
77 MHz Wide Node	201	76.5	74.6
66 MHz Thin Node2	224	56.2	56.2

Table 16. TCP/IP Switch Performance

It must be emphasized that the performance of the IP protocol is a strong and complex function of the characteristics of the network, the processor, the processor's memory bandwidth, and a long list of software-tuning parameters termed ″Network Options″. In Table 16, we list the measured latency and bandwidth, as this data shows the performance relationships between the SP switch and the several SP node types.

The latency and bandwidth are shown for the new SP switch. The uni-directional and bi-directional bandwidths are essentially the same. The performance is not limited by the switch speed but by the efficiency of the SP software and the switch adapter. For the SP switch, the relationship between processor and communication speed is strong. Looking at the two bandwidth columns for each processor, we see a substantial performance increase between a 66 MHz Thin Node2, and 77 MHz Wide Node. For each case, the processor performance is clearly the limiting factor.

The SP switch performance data given above are upper bounds to performance. In general, the latency for communication between parallel applications is higher, and the sustainable application bandwidth is lower because of system overheads and bottlenecks: Data destined for an adapter and data being recovered from an adapter travel through several software layers and share I/O buses with other I/O devices.

5.3.3 Virtual Shared Disk (VSD)

Virtual shared disk (VSD) is a software layer between an application and disk devices. VSD allows requests for data blocks to be resolved from locally attached disks or from disks attached to other processing nodes. When the data block requested exists on another processing node, VSD issues remote I/O requests across the SP switch. Currently, the primary application using VSD is Oracle 7.3 Parallel Server. The performance impact of VSD depends on the application and block size of the databases. OLTP applications may be concerned with additional CPU cycles, whereas decision support system (DSS) applications are more concerned with transfer bandwidth.

5.3.4 Sizing and Configuring a Control Workstation

The control workstation links to the supervisory subsystem of the SP system through the RS-232 asynchronous link. This link has to be provided for each SP frame to be controlled by the control workstation. Thus, if four frames are planned, the control workstation should be configured with at least four tty ports. This link uses a custom protocol for supervisory and tty data communication (point-to-point). The RS-232 operates at 19200 baud. The byte-oriented interfaces of the RS-232 will inherently put a heavy load on the CPU of the control workstation, especially at faster baud rates.

The control workstation is configured separately from the SP configuration. It must be an RS/6000 machine, and typically, it can serve 64 nodes. The RS/6000 model of this control workstation, though, is not easy to determine. It should have enough processing power to handle the demand of the software, the PSSP, and the hardware RS-232 lines interrupting one byte at a time. It should have the following I/O devices and adapters:

- Disk for diagnostics
- One RS-232 port for each SP frame
- Keyboard
- Graphics adapter
- Graphics monitor
- Tape/CD-ROM drive
- External LAN adapter

Usually, a LAN connects the control workstation to the nodes through the Ethernet. With the external LAN adapter, the workstation can also act as a gateway to external networks when no direct connections (FDDI, token ring) exist from the nodes.

If the control workstation also acts as a file server of some kind, enough storage space must be available. This depends on the type of server it is.

5.3.5 Sizing and Configuring an SP System

Since the initial introduction of the IBM RS/6000 SP systems, IBM has continued to announce new node types with faster performance and better price/performance ratios. In particular, a significant announcement has been the option of selecting SMP nodes for the SP system.

It is common to believe that with SMP nodes, everything will now run best on SMP nodes and uniprocessor nodes will never be needed again. This is most definitely not true. As will be described later, it is not as simple as that, and there are many factors that must be considered when selecting nodes.

It is also important to realize that the new SMP-based high nodes for the SP are aimed at commercial applications. The high nodes may not be appropriate for scientific/technical applications. Currently, the high nodes support TCP/IP traffic only over the SP switch and do not support parallel development software such as Message Passing Library, Parallel Environment, and PVMe.

5.3.5.1 Capacity-Based Node Selection

When selecting nodes for an SP system based on capacity, we can use the methodology shown in Figure 53.

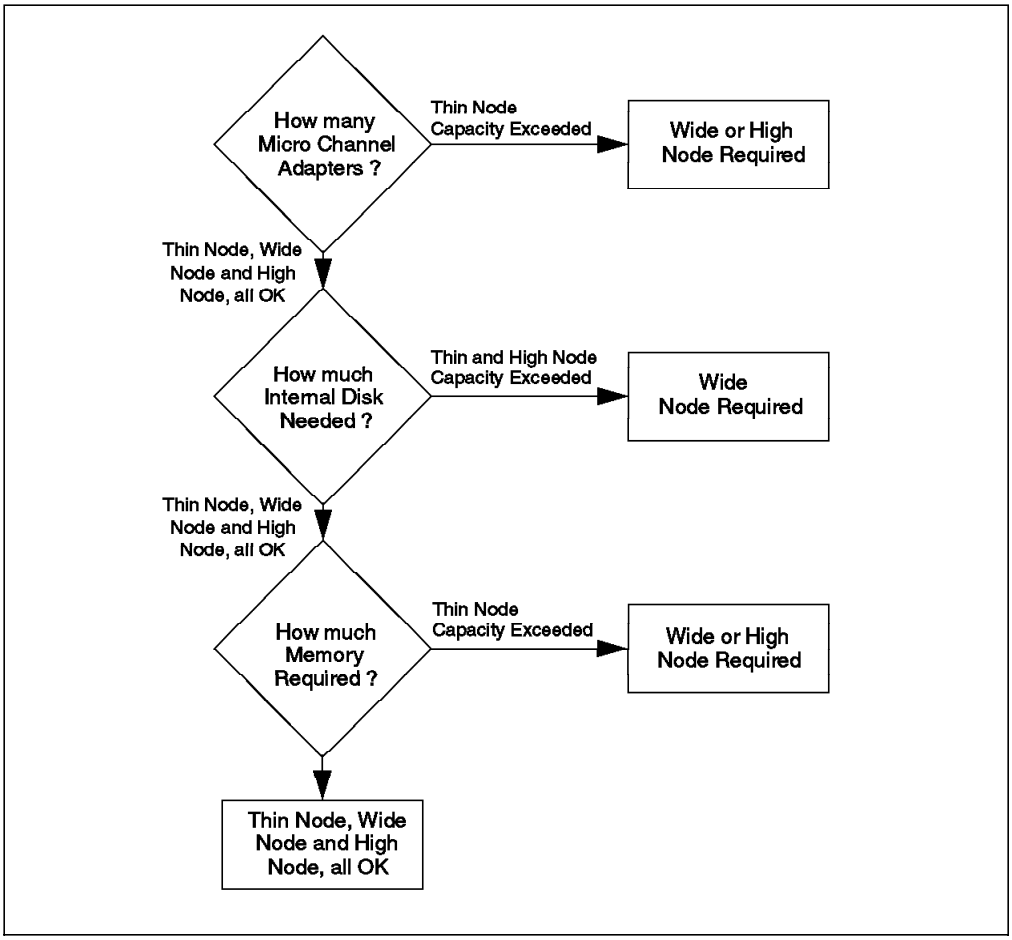


Figure 53. Capacity-Based Node Selection Diagram

First, look at the required capacity for adapters, disk and memory, because this will help you to decide whether a thin node has a sufficient capacity. As you

consider this, take particular care to include all Micro Channel adapters that will be required, both now and in the near future.

High Availability

Remember that for a high-availability solution, you will often need to have additional adapters for redundancy.

Remember that an internal disk is used differently for different applications:

- In the commercial world, it is usual to use the internal disk on each node only for the AIX operating system, paging space, application executables, and any temporary data. It is usual to store any critical customer data on external disks so that they can be ″twin-tailed″ to provide for high availability in the case of failure of the primary node. Internal disks cannot be accessed from another node if the node itself has failed.
- This is in contrast to some scientific/technical applications, where the volume of data is not as large and high availability is not so important. In these cases, sometimes only using disks which are internal to the node is the adopted solution.

Micro Channel Adapter Slots: Thin nodes have a built-in Ethernet adapter that can be used for the SP Ethernet (which is mandatory). Thus, all available MCA slots can be used as required. The wide nodes do have a normal Ethernet adapter, meaning one MCA slot will always be used for this purpose and one will be used for a SCSI adapter.

In many cases, each node will require an SP switch adapter; an alternate (user) network adapter, for example an Ethernet or token ring adapter; and some kind of disk adapter (SSA or SCSI). In practice, the thin node slots can fill very quickly, and they are almost never appropriate where high-availability solutions are required (HACMP).

Internal Disk Capacity: Internal disks will normally at least contain the AIX operating system, paging spaces, any temporary files, and possibly application executables. For many commercial customers, their real data will be stored on external disks (e.g. SSA or RAID) outside the SP nodes.

Maximum Memory Capacity: As there are many choices and combinations of memory cards for SP nodes, please check with the RS/6000 Configurator Program.

5.3.5.2 Performance-Based Node Selection

To consider performance of the various nodes, we will need to discuss particular application types. There will be a different set of factors to consider in each case.

Obviously, this chapter cannot discuss all applications that are available on the SP system. We need to differentiate between serial applications and parallel applications as we look at performance of the various SP nodes. For this reason, we will look at some of the most common applications and database systems running on the SP system today. These examples will help make you aware of the factors to consider when making the decision.

If we are looking at a single serial application on the SP system, by definition, it will not utilize more than one node within the SP. It is very common to run numerous serial applications on the same SP system, either as server consolidation applications, or as client/server applications. In the case of server consolidation applications, there are a number of different applications running on discrete nodes within the SP, usually to achieve reduced management and support costs. In this scenario, we are not running anything in parallel, and we may choose to use a database (RDBMS) system, but we will be running a serial or classical version.

Such applications are likely to be good contenders for running on a high node, as long as the particular application has been designed and written to exploit an SMP architecture. In the commercial world, with typical database applications, this is very often the case but not always.

For each application that we may choose to run on a high node, we must check that it will exploit the SMP architecture. Otherwise, we just end up with only one processor out of eight (in an eight-way SMP node, for example) being utilized. If an application typically only scales to a four-way SMP system, it will not take advantage of a six-way or eight-way system.

The high nodes are not intended to run scientific/technical applications. The floating-point performance and price/performance ratio of the POWER2 uniprocessor nodes are superior to the PowerPC 604 performance for such applications. The POWER2 nodes will be the obvious choice for these applications. In practice, a technical or scientific environment might well have a requirement to run particular commercial workloads, such as file servers or communication gateways, and these commercial workloads (which are not floating-point workloads) may be run on SP nodes preferred for commercial applications.

For commercial parallel applications (applications written to exploit more than one node within the SP system) there is even more to think about. We first need to find out if the application supports an SMP architecture (each node should be an SMP), but also, we need to make some decisions on how we might mix nodes within such a parallel application.

If we were starting from scratch, for example, we might choose to implement all nodes as high nodes (assuming that our parallel application does support SMP). We would probably select nodes all of equal power and similar configuration for our parallel application. This may well be a good solution. But what would we do if we already had a number of POWER2 thin nodes running this application and we wished to add more power by adding high nodes (if the application exploits the SMP architecture). This would lead us to implement different types of nodes, which may well lead us to have an increased management/administration overhead.

We could also think of a scenario where we start off with a serial application (for example, the database server for an SAP solution). This serial database might run on one high node as we grow from two to four, to six, and eventually to eight processors within this one node. At this time, we might find that we need more performance and we choose to implement a parallel database to give us an additional growth path. However, what would we add to the existing configuration of one large high node? We might initially consider one more two- or four-way high node, but this would give us a performance imbalance that we would rather avoid. In this scenario, technically, a good solution would be to

add an additional eight-way SMP node, but cost-wise, this is likely to be too large a jump for the customer. An alternative might be never to grow the first node to an eight-way SMP but to introduce a second node earlier, when it was probably still a four- or six-way node.

This chapter cannot give you all the answers to such questions, but you do need to be aware of such issues as you design your SP solutions with your customer.

The following flow diagram can be used to help make the right choice for nodes to select from a performance point of view. It is assumed that you have already made a judgment about nodes on the basis of capacity, as discussed earlier.

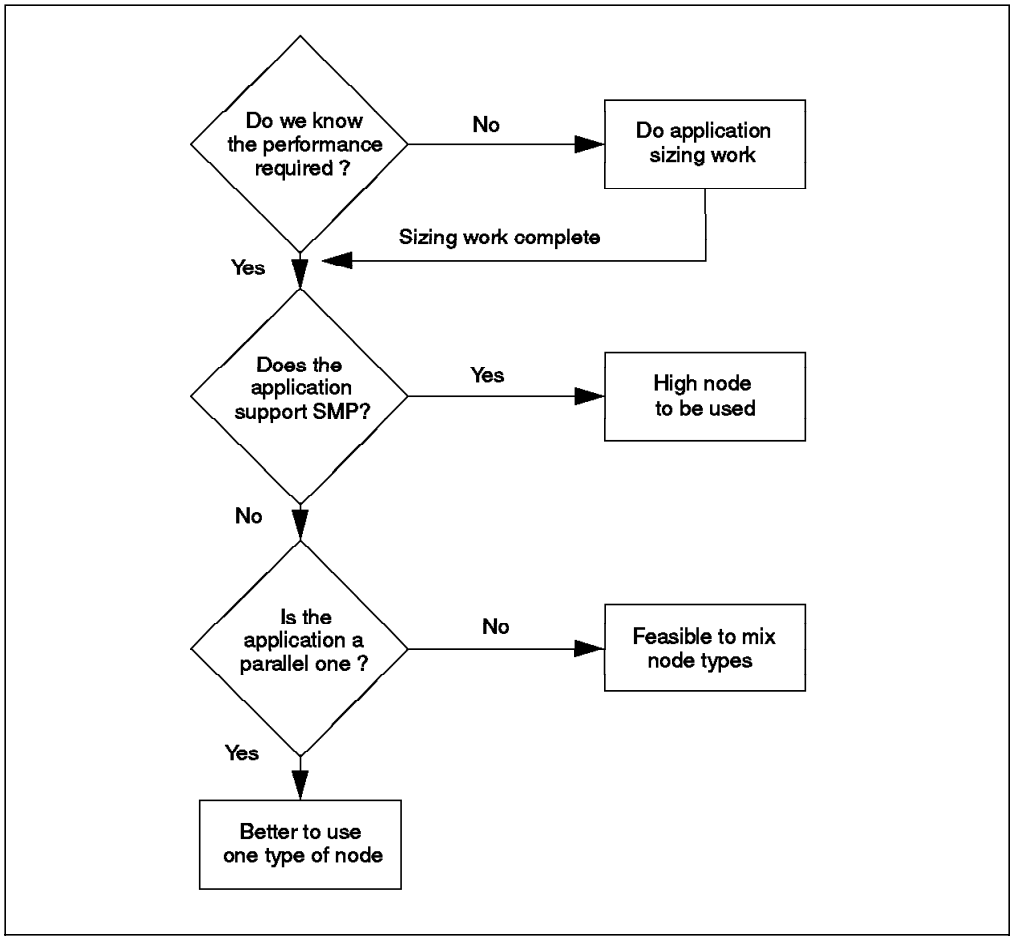


Figure 54. Performance-Based Node Selection Diagram

We often talk of scaling of performance within the SP system. How well any particular application scales as we add SP nodes will clearly depend on many factors. One factor that will help us evaluate the scaling will be to examine the ratio of the CPU computational workload as compared with the communications workload. As we add nodes to the SP, we would like to evenly distribute the work across the nodes, and how well we can achieve this will depend on this computation/communications ratio.

One factor which will be important, particularly when we are running parallel applications, is the way applications such as parallel databases can handle nodes of varying power within the total configuration. High nodes within the SP have just recently been announced, and some questions cannot be answered at this stage, but some information is supplied in the following sections. More

performance testing and customer experience will soon give us a much clearer indication in this area.

The factors looked at so far are exploitation factors as opposed to price/performance factors.

In the following, we will briefly look at some price/performance indications that will show us that the price/performance of the current nodes is pretty similar. This means that we can select the most appropriate nodes for a particular application based on the best choice for the customer, rather than picking out particular nodes purely because of price/performance.

5.3.6 Server Consolidation (Serial Applications)

In a server consolidation environment, we will not be running parallel applications, so the most important consideration is whether the particular applications to be implemented support SMP processors and will take advantage of such nodes. In some cases, we find that an application does exploit an SMP system but only to a certain extent. Maybe it can take advantage of a four- or six-way SMP system but not an eight-way.

In general terms, server consolidation applications are likely to be among the best ones for running on high nodes within the SP. Adopting high nodes will allow us to minimize the number of nodes required, but assuming that we wish to isolate applications, we can still have enough nodes within the SP to allow for some level of potential redundancy from an availability point of view.

One of the issues when running server consolidation applications is the performance available on any one SP node. By definition, server consolidation applications will not be able to utilize more than one node, and therefore, as the application grows in performance terms, the maximum performance available on any individual node can be a deciding factor.

The introduction of the high nodes is a major advantage now, in that the customers can grow their applications from the smallest of uniprocessor nodes right through to an eight-way high node with greatly increased performance. The binary compatibility of AIX running on each node means that the same application can run unchanged as we move from one node to another.

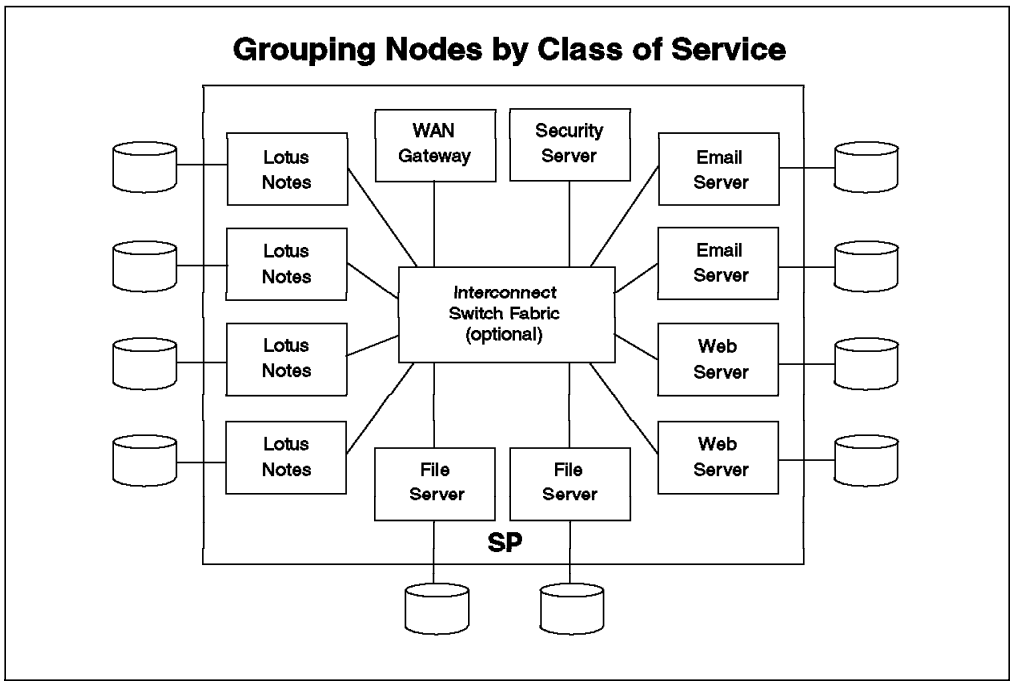


Figure 55. Server Consolidation

5.3.7 Parallel Sizing Factors

Sizing a scalable system for commercial applications is different from sizing a non-scalable system. It depends on the type of commercial applications, and the following factors should be considered. Usually there are two types of commercial applications predominately used on a scalable system: decision support system (DSS) and on-line transaction processing (OLTP).

The performance metrics for the system designed for OLTP applications are:

- Throughput
- Response time
- N-node scale-up

The performance metrics for the system designed for DSS applications are:

- Throughput
- Response time
- N-node scale-up
- N-node speed-up
- Database scale-up

5.3.7.1 Scale-Up and Speed-Up

In the case of scale-up, twice as much hardware capacity typically should perform twice the work in the same elapsed time. Scaling up a scalable system can be done in two ways, as shown in Figure 56 on page 135. The first case occurs when increasing the data and resources equally. In this case, perfect linear scaling will produce the same elapsed time. In the second case, the amount of data increases while the resources stay constant. In the latter case, elapsed time should increase proportionally to the workload increase.

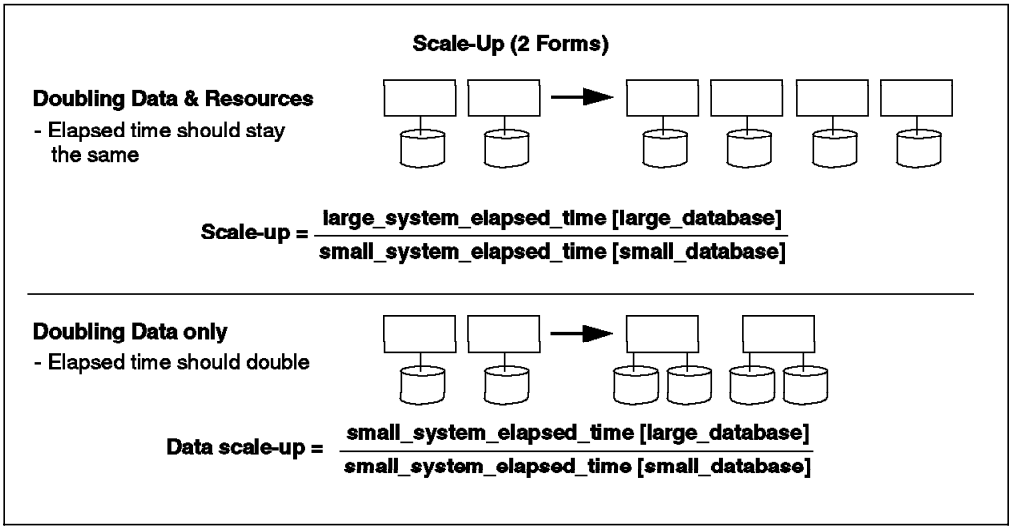


Figure 56. Scale-Up

In the first case, the ratio of resources to workload remains constant as more data and more CPUs are added. This is the primary reason for the existence of distributed memory parallel machines. Once the ratio of ″CPU speed to disk″ is ascertained, the parallel machine can grow to any possible database size, assuming perfectly linear scale-up (100 percent scale-up). Simply stated, twice the hardware and twice the data execute in the same elapsed time if scale-up is linear.

In the second case, the workload is increased while the resources remain constant. Increasing the workload within a single processing unit will scale favorably until a threshold is crossed in one or more of the resources. At that point, the elapsed time degrades significantly. When only the data is increased, computers rarely scale up linearly. Typically, a system will handle the workload up to a point, and then the phenomena of ″thrashing″ occurs. Once thrashing begins, efficiency decreases and more resources are required to achieve acceptable performance.

Most commonly, scale-ups are required in operational OLTP, interactive client/server, and batch applications. Thus, if we know that one CPU can support 200 end users, we would like to be able to combine four CPUs to handle 800 end users. If a system cannot scale up, the programmers may have to rewrite portions of the application or split it across two systems to support a larger number of end users. The objective of scale-up is to increase workload capacity primarily through replication of resources.

Speed-up is different. In this case, our goal is to use twice as much hardware to perform the same task in half the time. Typically, applications that require speed-up are strategic analytical applications such as data warehousing and data mining. Allowing end users to submit twice as many problems to be solved per day allows them to refine the accuracy of the result and meet deadlines. Inefficient speed-up results in fewer transactions per time period, slowing down the overall pace of the business productivity. The objective of speed-up is to improve response time to end users primarily through partitioning of workloads across resources. Speed-up is therefore more of a software concept that relies upon the underlying hardware to scale up. Thus, the industry speaks most often about scalability when comparing computer systems of any size.

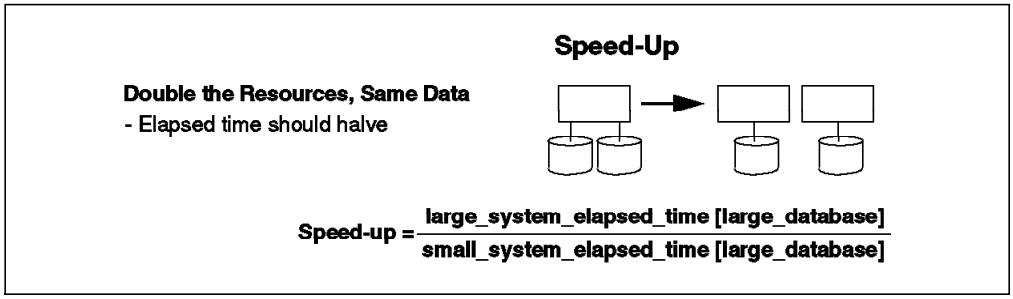


Figure 57. Speed-Up

According to Gartner Group, a system must achieve better than 80 percent linear scaling to be considered scalable and useful. This means achieving better than 80 percent scale-up or speed-up across a wide variety of production workloads. Since it is not possible that every conceivable workload will scale in this range, the best balance is achieved when most common production workloads run at better than 80 percent. While some may run at efficiencies of 60 percent or even as low as 40 percent, occasional low efficiency is tolerable. Consistently low efficiency is not tolerable.

5.3.7.2 Amdahl's Law

The speed-up factor on a parallel machine depends on what portion of jobs or subtasks can be parallelized. It can be described in the following diagram.

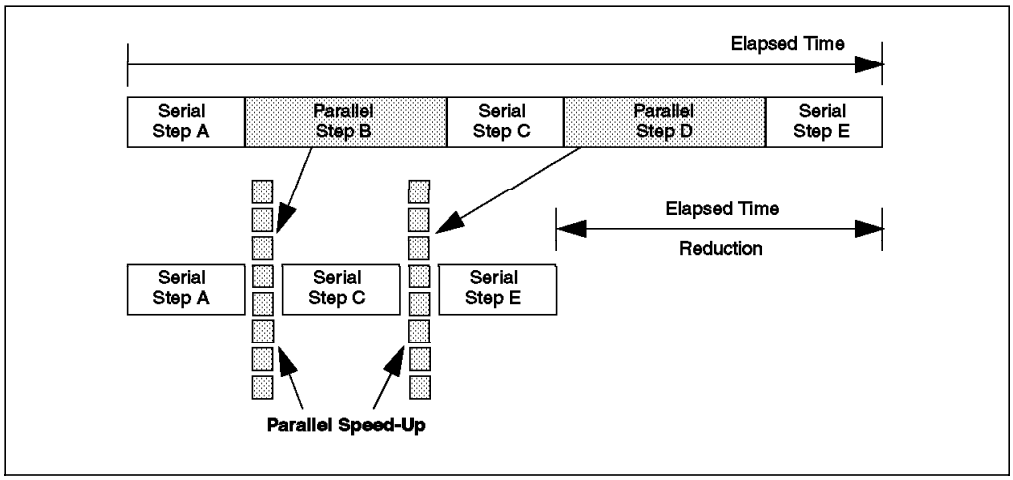


Figure 58. Amdahl's Law

In the above diagram, we see that some tasks can be parallelized. This will speed up the whole process. However, the performance is always limited by the slowest component because we can not parallelize everything in the parallel machine.

In distributed memory parallel machines, the most common reason for not achieving 100 percent scale-up or speed-up is a large serial processing component in the workload. A serial activity, for example, would be the emission of a million data rows to a LAN client. While retrieving the rows from the database can be done in parallel, the emission to the Ethernet or LAN cannot. Therefore, a major portion of the elapsed time is controlled by a low-speed, serial activity lowering overall efficiency.

5.3.7.3 Parallel Databases

There are two types of RDBMS supported in the SP system, shared disk systems (I/O Shipping) and distributed disk systems (function shipping). Each of these types has a different approach and implementation to take the benefit of the parallel function on the SP system. We will look at both of these systems in order to understand sizing requirements and node types that can take advantage of the parallel database architecture.

Shared Disk Systems (I/O Shipping): Oracle Parallel Server (OPS) is a parallel RDBMS using the shared disk system. For serial applications (not parallel), the Oracle RDBMS is already running on the RS/6000 SMP systems and has been for some time. In other words, it exploits an SMP architecture. Therefore, for any serial applications, if a high node is selected for performance or other reasons as described in this chapter, we are likely to find the high node a good solution.

As we consider parallel applications, we can see that a parallel database consisting of a number of high nodes running Oracle also is likely to be a good solution, if high nodes are required.

The option of running a parallel database with four eight-way high nodes, rather than 16 POWER2 wide nodes, for example, is likely to be the preferred option, as we will have far fewer nodes to maintain, manage, and administer. The situation is not quite so clear if we were to mix nodes, for example wide and high nodes within an Oracle parallel database solution.

How easy would it be to make sure that each node took its fair share of the workload according to its potential performance? Oracle 7.3 is well known to have an intelligent optimizer. It notes the fact that certain nodes get through their work more quickly than others, and it adjusts subsequent workloads accordingly. In this scenario, we would expect the high nodes over a period of time to accept more of the workload when compared to other, less-powerful nodes. Further experience with high nodes in real life -- and seeing how well they may work with such an optimizer that builds up a history of node performance -- will give us more detailed information in this area.

When running an Oracle parallel database on nodes that include high nodes, we will not need to run multiple Oracle instances on each node. We will run one instance of the Oracle software, and it will take into account the multiple processors and manage the resources.

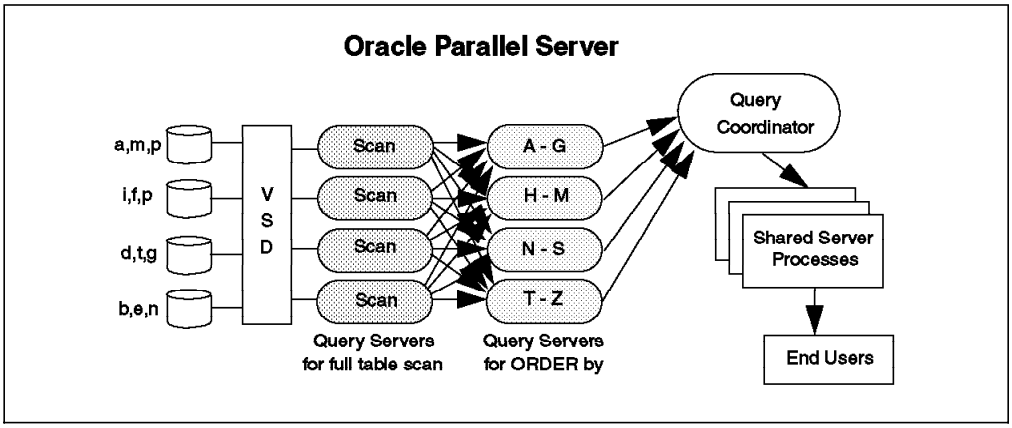


Figure 59. Typical Oracle Parallel Server Implementation

Distributed Disk Systems (Function Shipping): There are three RDBMS using distributed disk systems: DB2 Parallel Edition (PE), Informix XPS, and Sybase.

In the case of a serial database, the IBM RDBMS software, DB2/6000, is the choice to run on SMP systems and can exploit a high node within the SP. For serial applications, such as those within a server consolidation environment, high nodes may well be a sensible option.

In the case of a parallel database, DB2 Parallel Edition works in a somewhat different way than the Oracle Parallel Server.

DB2 PE partitions the data and spreads it across the nodes within the parallel database. Again, as with all parallel databases, it is the best suited solution to use equal power and similarly configured nodes across the entire environment. We would also in this case expect to place volumes of data in such a way that each node is kept equally busy.

As with Oracle, DB2 PE will work reasonably well with a parallel database that is running on high nodes, but there are some differences.

Any specific query in the case of a complex query (in a decision support environment, for example) cannot be split across the processors within a high node. To achieve this level of parallelism within the node, we would need to run multiple instances of the DB2 software. In addition, we would need to make sure that appropriate volumes of data were placed on each node to give us suitably balanced performance in line with the node performance.

There is clearly administrative overhead in running these multiple instances. For this reason, it may be better in some cases to utilize only uniprocessor nodes because this results in only one type of node within the parallel database. Once again, time will tell us which options will be best in particular circumstances as we get more information and experience with the high nodes in a parallel environment.

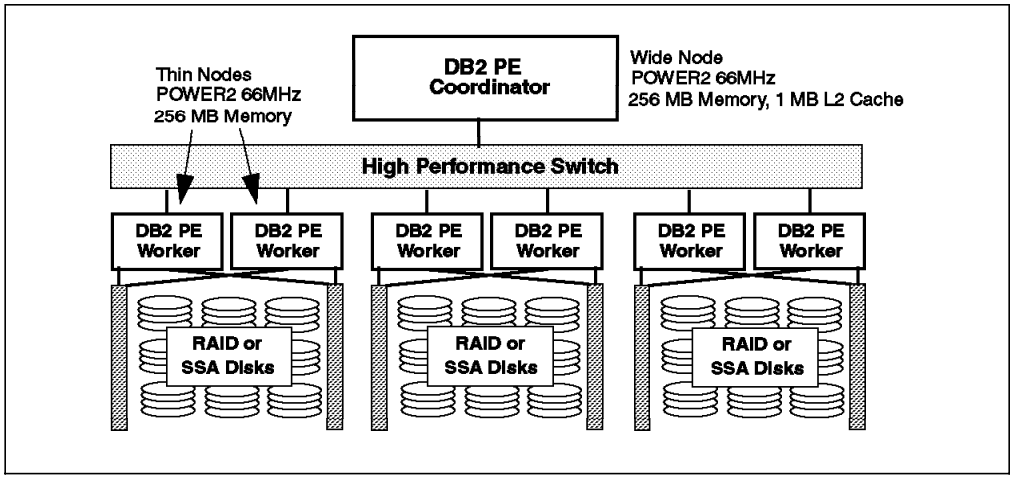


Figure 60. Typical DB2 Parallel Edition Implementation

5.3.8 Decision Support System (DSS)

DSS or data warehouse applications are characterized by very complex query transactions against large databases with usually few users. Normally, the users run ad hoc operations, and transactions vary greatly in size, frequency and complexity. Applications that belong to this category are ad hoc analysis, profitability, credit card use, target market analysis, cross selling, and inventory management. The response time for this kind of application can be very long (minutes and hours). The industry-standard benchmark for the DSS workload is TPC-D. For detailed information about TPC-D, please see 6.3.4, “TPC-D” on page 166.

To achieve the goals of DSS developers, the twin strengths of scale-up and speed-up are required. Speed is needed to permit end users to solve business problems in minutes and hours instead of overnight, as in the past. Fast response times, even for complex queries, permit the user to issue multiple refinements to a query. If unusual results occur, fast response time is needed to further explore the anomalies or surprises in order to guarantee the results are accurate.

The second important thing of the DSS is scalability. As more and more data is accumulated in the database, it is desirable to maintain the same response times to the commonly used queries.

Figure 61 is one example of a parallel data warehouse speed-up as well as scale-up. Using DB2/6000 PE, a full table scan was invoked.

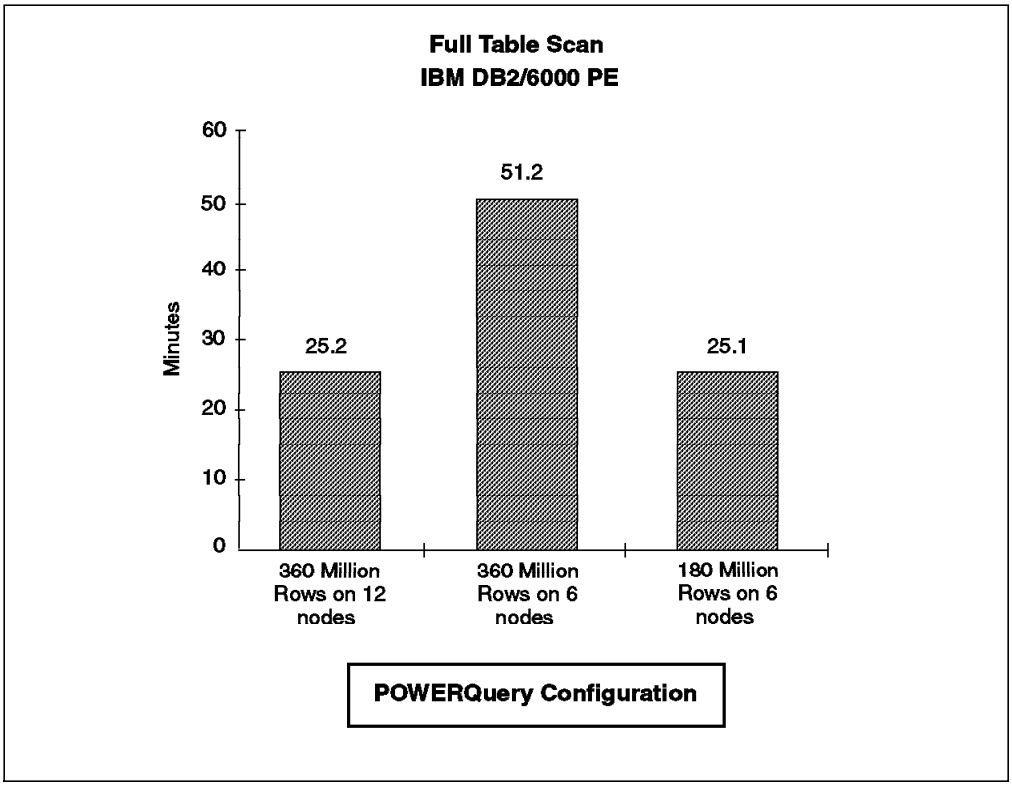


Figure 61. DB2 PE Test Result

In the first and second bars, speed-up is shown. In this case, the workload is constant at 360 million records scanned. By doubling the number of processor nodes in the IBM RS/6000 SP, the response time is cut in half. Scale-up is

shown in the first and third bar. In this case, the workload per node remains constant: The number of data rows and the number of nodes were both doubled. The result is a nearly identical response time, ensuring the end users receive consistent performance when expanding the database size. Once the proper ratio of data to computation power is known for a given data warehouse or DSS, scalability ensures that nodes and data can be added indefinitely in that ratio.

Nodes Sizing: In the decision support applications, we usually will either be running the database on a single node (in serial) or in parallel if the most powerful SMP node is not powerful enough. There will be a point where any SMP solution will not be powerful enough to support a large DSS system, probably when the database volumes run into hundreds of gigabytes, because of I/O contention at the single node. At this point, a parallel system will be required, and this could include uniprocessor nodes or high nodes. It may be better to avoid combinations of nodes, where possible.

With regard to the SMP processors, we would expect DSS performance and scaling to be good. As we combine processors within an SMP system, it is usual to get a diminishing return on the investment as we add additional processors. This is to be expected for any SMP system. However, the scaling factor for DSS workloads is likely to be better than for OLTP (TPC-C).

In a TPC-C workload, a large proportion of the time is spent performing kernel activity, resulting in a certain level of operating system locking. In contrast, we would expect a DSS workload to typically spend more of its time in the applications (user activity as opposed to kernel activity), and this would normally result in better scaling. This is assuming that the application does not introduce an additional level of locking.

For this reason, a high node running a TPC-D workload within the SP is likely to display better scaling as we grow from two to eight processors than a high node running a typical TPC-C workload. Such factors will, of course, depend on the actual application or database in use.

5.3.9 On-Line Transaction Processing (OLTP)

OLTP applications usually demand rapid interactive processing for a large number of simple transactions. Normally, an OLTP transaction has no floating-point calculations and has poor locality of code and data. OLTP applications require a fast, predictable and consistent response time to satisfy user workloads. The industry benchmark for OLTP is TPC-C. (Please see 6.3.3, “TPC-C” on page 165.)

In the parallel machine, data must be partitioned to enable scale-up, and the programming model must change accordingly. Usually, an OLTP machine has three major functions: application, transaction and database manager. Sometimes, in a small OLTP environment, there is no need for a transaction monitor (transaction router) to simplify the configuration. In a parallel machine, each of these parts runs on one or several nodes and communicates to each other through high speed network or LAN. Figure 62 on page 141 shows the simplified diagram of OLTP implementation on a parallel machine.

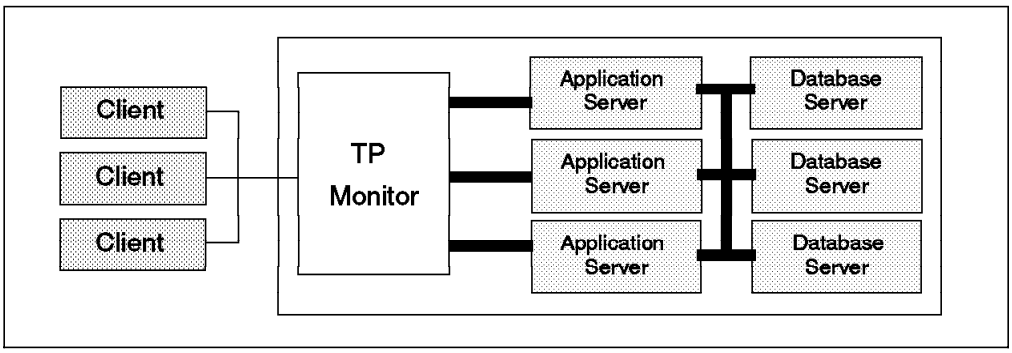


Figure 62. Typical OLTP Implementation Diagram

A transaction processing (TP) monitor such as Tuxedo or CICS/6000 allows the application to invoke a transaction on any node in the network based on control information from the client station. Thus, a transaction invoked by the client may contain data that the TP monitor uses to route the ″run transaction″ invocation to a node considered optimal by the application designer. This added sophistication is needed only when the OLTP transaction rate is extremely high and the response times must adhere to strict, subsecond performance. This is because the ″routed transaction″ gains less than 100 milliseconds of response time, in most cases.

In the above diagram, the bar between the application server and database server is the interconnection fabric. In the SP machine this bar represents a message-passing event, running between 50 - 200 microseconds, depending on the vendor(s). While this seems like a dramatic difference, in effect, it changes the response time for the end user by less than 1/5 of a second, if that much.

The TP monitor logic that does the routing can be placed in a node of the parallel machine or the client station. The partitioned database servers may be separate database instances or may appear as a single database to the application. To achieve a single image database, a parallel database manager is required.

Nodes Sizing: In most situations, OLTP is likely to be reasonably well-suited to run on SMP nodes. We would usually expect a lot of users who are each running relatively light transactions (when compared with complex transactions in a DSS, for example). This is more suited to an SMP environment, where we can spread the transactions across the CPUs more easily. In effect, OLTP workloads have an element of parallelism built into them as a result of the fact that we typically have a large number of relatively light transactions. In each application case, it is important to investigate whether the application in question will support SMP processors.

One of the factors that will significantly effect scaling of performance across nodes in a parallel database on the SP will be the amount of I/O that is performed on disks attached locally (on the actual node that requests the I/O) as opposed to I/O to disks that are remote from this node. Clearly, the more disk activity that is local, the better the scaling. TPC-C transactions, for example, tend to perform less than 15 percent of their I/O remotely. In contrast an application performing 50 percent of its I/O to remote disks would not scale so well. Such factors need to be considered as we size our SP system for specific applications.

5.3.10 Scientific Applications

Scientific applications depend on single processor performance (integer, floating-point, and memory subsystem) of each node to achieve a good scientific performance on the SP system. Another important factor is the communication subsystem between nodes. It must have a low Message Passing Interface (MPI) latency, high bandwidth and good TCP/IP performance. A balanced system can produce a good application-level performance.

Several benchmarks are used to measure the throughput of scientific applications on parallel machines, including LINPACK HPC, NAS, and SPECrate.

The performance result of the NAS benchmark can be seen at URL address:
`http://www.nas.nasa.gov/NAS/NPB/index.html`

The performance result of the LINPACK HPC can be seen at URL address:
`http://www.netlib.org/benchmark/performance.ps`

The performance result of SPECrate can be seen at URL address:
`http://www.spechbench.org`

Sizing Nodes: When selecting node types for a typical scientific/technical application, we will usually have less of a problem in terms of which nodes to utilize within the SP system. These applications will not exploit the high nodes and will need the higher floating-point performance of the POWER2 nodes. The individual PowerPC 604 processors do not have high floating-point performance and would not be used for scientific/technical applications. The user space protocol is not supported across the SP switch when using high nodes. This is another reason why high nodes are not suitable for scientific/technical applications. Therefore, combinations of POWER2 uniprocessors are likely to be the best solution. However, in a scientific/technical environment, there is often a requirement for commercial type workloads, such as file servers, print servers and other such servers for which an SMP node can be used.

5.3.11 Conclusion

The IBM SP system provides a scalable solution that can be used for commercial, technical and conventional (serial) applications. The scalability of the SP system depends on the type of applications (commercial, technical). The SP switch has shown very good performance with a native user space protocol or TCP/IP.

5.3.12 References

`http://www.rs6000.ibm.com/hardware/largescale/index.html`

`http://www.rs6000.ibm.com/resource/technology/tech_downloads.html`

`http://www.rs6000.ibm.com/resource/technology`

5.4 SP and SMP Selection

Sometimes, there is some confusion in choosing between an SMP and SP system for the multiprocessor machine. They both have their advantages, and the guidelines below give considerations and background information on when to implement an SMP or an SP system. The discussion is based on the commercial processing environment. IBM SMP machines utilize PowerPC 601 or 604e and do not provide better floating-point performance than POWER2, which is very important for scientific and engineering applications. Throughout this chapter, the SP will be referred to as a distributed memory processor (DMP) system.

5.4.1 SMP and SP Considerations

In order to differentiate the SMP and DMP systems, we have to understand the potential bottlenecks on both.

Memory Bandwidth: This relates to the question of whether the system has enough bandwidth for the main memory/memories to do the work.

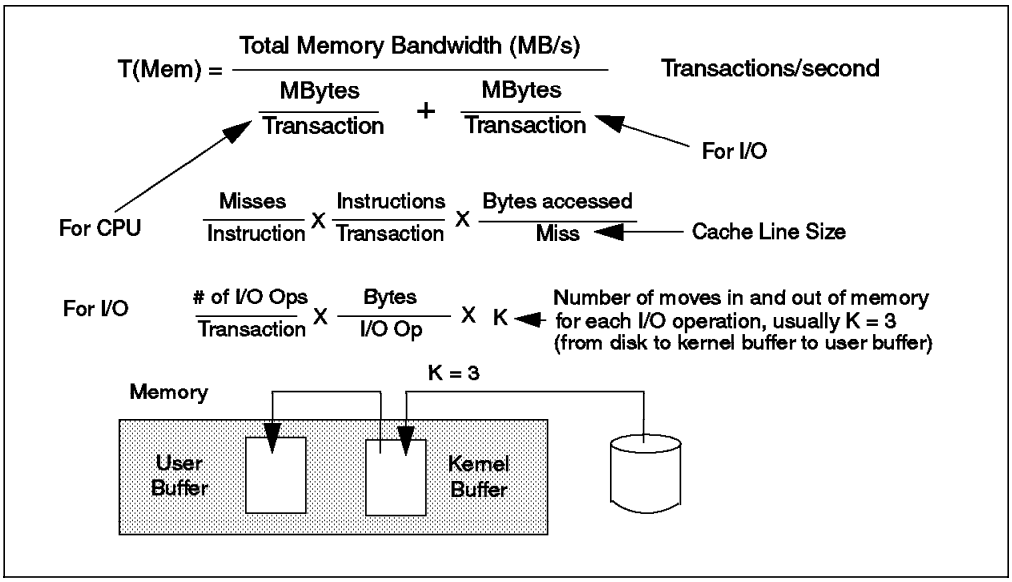


Figure 63. Memory Bandwidth

Processors: This relates to the question of whether the processors have enough cycles to do the work.

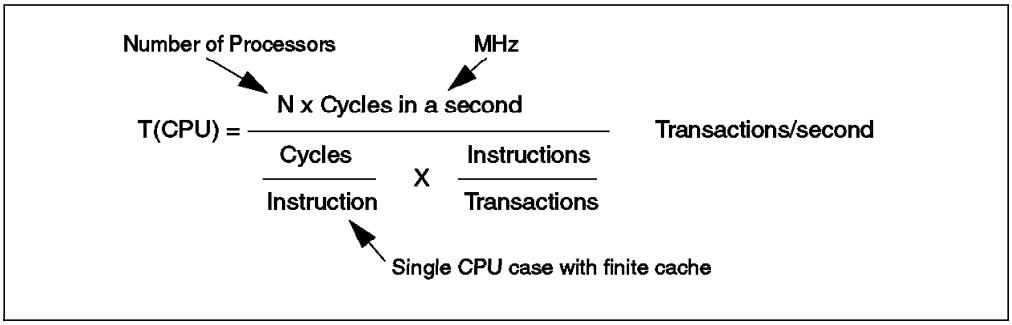


Figure 64. Processor

If we assume that the CPUs have the same basic design, then T(CPU) will generally be slightly lower for DMP machines than for SMPs because internode communication overhead will make the path length on DMP somewhat longer than on SMP.

Switch Bandwidth: This relates to the question of whether the system has enough switch bandwidth for communication and remote I/O.

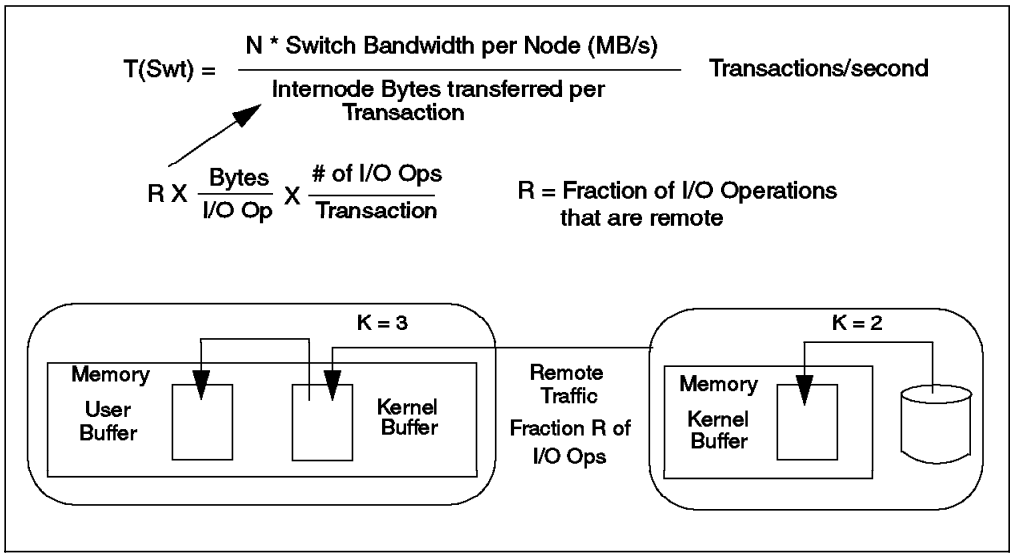


Figure 65. Switch Bandwidth

In this case, remote I/O operations also involve additional traversals through memory, which must be taken into consideration for local memory bandwidth.

Impact of Bottlenecks: As a comparison between SMP and DMP systems, we try to calculate the throughput (transactions per second) based on previous equations. These calculations are based on these assumptions:

- N = 16 which can be 16-way SMP or 16 uniprocessor nodes in a DMP machine
- CPU cycle is 150 MHz with a 128-byte line size
- Memory bandwidth is 1.5 GB/s (total for SMP and per node for DMP)
- Switch bandwidth is 50 MB/s per node (for DMP)
- I/O can be configured to not be a bottleneck in both SMP and DMP cases
- Instruction/transaction is 3.3 million (M) for SMP, 4 million for DMP (TPC-C like characteristics)
- 3 cycles/instruction (single CPU, reasonable size L1 and L2)
- Cache miss ratio is 15% (commercial type)
- 150 I/O operations per transaction
  - Each of 4 KB (0.004 MB)
  - 50% local, 50% remote for DMP
  - 40% hit in I/O cache (K = 2 for local or 4 for remote), 60% from disk (K = 3 or 5)

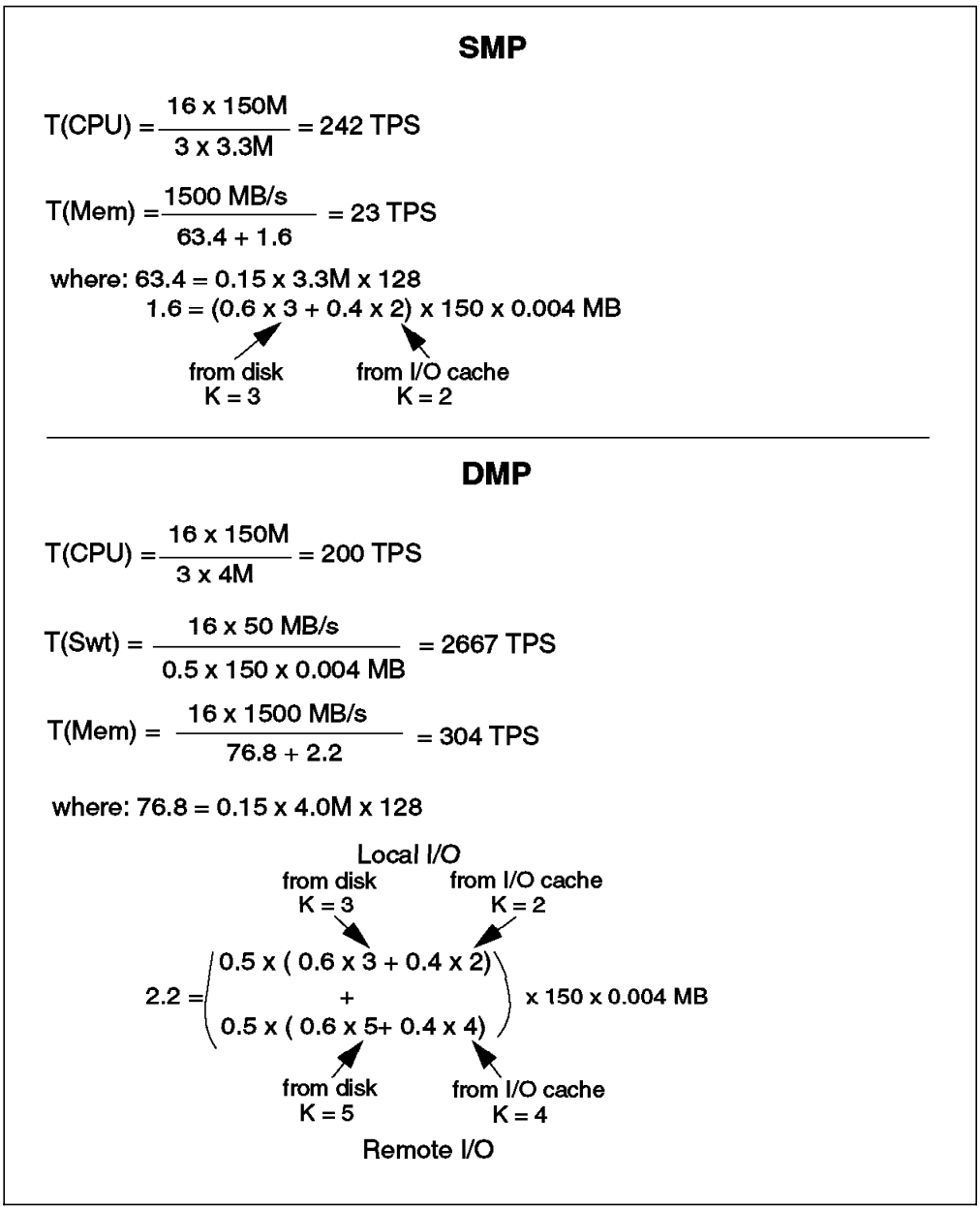


Figure 66. SMP vs. DMP Throughputs

From the above calculations, we can see that the SMP system has lower memory bandwidth throughput than the DMP system. Many other factors in SMP and DMP (such as software inefficiencies) are not modeled here. Meanwhile, on the DMP system, the CPU is the ultimate bottleneck.

In most cases, an SMP system is cheaper than a DMP system, but the SMP system has certain limitations compared to the DMP system (scalability, expandability). In order to explain this, IDC uses a simulation model of the interrelationships between hardware components implementing data warehouse application to estimate the price of a balanced system regardless to the number of users on the system. The model expects the following ratios to exist within any hardware system implementing a strategic business application (such as data warehouse):

- 3 GB of disk per GB of raw database data

- 35 GB of disk per I/O processor (IOP)
- 3 disk drives per I/O processor
- 105 SPECint92 rating per I/O processor
- 9 SPECint92 per GB of database data
- 0.73 MB of memory per SPECint92
- 0.63 MB/s of bus speed per SPECint92

It seems that the SMP system can provide the solution for these requirements, but as the problem size grows, there is a point where the SMP system will not be able to provide a good, balanced system. In general, the following factors are SMP limitations.

- I/O Bandwidth

  This factor is related to how fast the system can move data from disk to memory. With the DMPs, the number of IOPs will scale with the number of processor nodes. Best-case analysis is that system I/O can run up to system bus speeds, which currently are in the 1 GB-per-second range. In this case, SMP systems based on the model would break down with about 600 GB to 700 GB of data (assuming 20 MB/s IOPs and assuming each IOP controlled 35 GB of disk). A more realistic number might place the I/O bandwidth at half the bus speed, in which case, a 300 GB database would stress the system.

- Memory Size

  This factor is related to how much physical memory will be needed to support the applications. For 32-bit architectures, maximum memory is 4 GB or 2 GB, assuming operating system overhead. With the DMPs, the memory size scales with the number of processor nodes. Under perfect conditions without operating system overhead, the model would stress an SMP with a database size of 600 GB to 700 GB. Under more realistic conditions, the maximum database size drops between 300 GB and 400 GB. With 64 bit systems the value will change dramatically, with terabyte databases being manageable.

- System Bus Bandwidth

  Most SMP architectures are system-bus limited. With DMPs, system bus bandwidth scales linearly with the number of processor nodes along the local memory to processor dimensions. It scales logarithmically with the number of processor nodes for remote memory access (i.e. remote memory access does not improve at the same rate that processors are added). By assuming system buses running at 1 GB/s, SMPs become stressed under the model with databases in the 200 GB range.

Another way to address the same problem is to look at the processors based on SPECint92 required to drive and IOPs (the model assumes 105 SPECint92 per IOP). It is also assumed that the estimated number of 12 processors per SMP is considered a maximum CPU target for system architects to balance various SMP features (bus speed, memory, CPU speed) to run general-purpose workloads. If processors are running at 100 SPECint92, then the SMP runs out of processors with databases in the 100 to 200 GB range. Setting the SPECint92 to 200 allows SMPs to run approximately 300 GB databases. This is a linear relationship, so every 100 SPECint92 leads to about 100 GB increase in database size. The above analysis provides evidence for a 200 to 300 GB limit on SMPs under ideal conditions.

In summary, the general cross-over point between SMP and DMP systems is dependent on several variables:

- Size of the database and the largest tables
- Number of concurrent users (usually 50 to 200 concurrent)
- Complexity of database queries (from I/O only to extreme CPU demand rates)
- Speed of the system bus, microprocessor, and I/O subsystem
- Amount of data actually accessed by each query (hundreds of rows to millions of rows)
- Definition of acceptable end user response time (minutes or hours)
- Quality of the RDBMS, algorithms, and partitioning capability
- Database size and user population growth expectations (usually 2 - 4 times over 2 - 4 years)

Large data warehouse applications with few users and simple queries can be managed on an SMP architecture. Conversely, even small data warehouse applications with a large population of users running complex queries can strain an SMP beyond the breaking point. IBM considers that between 100 GB and 200 GB of actual data, the data warehouse customer must seriously begin considering a DMP solution. Therefore, it is important to know beforehand what the end users will try to accomplish with the data warehouse application. Since this is rarely the case, the implementors are advised to assume a tripling or quadrupling of actual data within three years of initial installation. Note the emphasis on actual data. Actual raw user data should usually be multiplied by three to estimate the total disk capacity needed in a system. This accounts for spool and sort space, indexes, data load staging areas, and operating software overhead. Therefore, a 200 GB database will be installed as 600 GB of real disk capacity.

Chapter 6. Benchmarks

It is very important for customers as well as for computer manufacturers to compare the performance of different computers. However, it is very difficult to find an absolute measurement because, nowadays, computers are complex systems in which many components influence the overall performance of the system. System performance is especially dependent on the kind of application software that is running on the system. Moreover, benchmarks are necessarily abstract and simplified models of all those environments. For this reason, benchmarks represent a good yardstick for comparing different systems rather than a precise tool for capacity planning for a given customer application environment. No benchmark can fully characterize the performance of a system in a true production environment:

- The behavior of benchmark applications is essentially constant on a given system. Real applications, when executed several times, almost invariably have different inputs, and consequently exhibit different behavior each time.
- Benchmarks are executed under ideal circumstances. The benchmark is typically the only application that is executing on a system dedicated to a single user. For this reason, system overheads, such as paging and context switches, are lower than in actual production use of a processor. Benchmark processors are often equipped with the “latest and greatest” memory and disk subsystems, features that may not exactly match a system that is of interest to a customer. In this sense, benchmarks represent an “upper limit” to system performance.
- Frequently, benchmarks are specified algorithmically (“pencil and paper” specification), or do not place restrictions on the amount of tuning that can be performed. In such situations, the application programming skill that is available to a vendor can play an extremely important role in determining the performance measured for the benchmark.

Nonetheless, some insight into the performance of a computer is provided by a benchmark:

- A computer that performs well on all benchmarks in a given class such as floating-point-intensive codes with data structures that are too large to fit into cache memory is likely to perform well in all applications that share these characteristics. Customers are usually well-informed of the characteristics of their primary computational load.
- A processor that performs well in “throughput” benchmarks -- one in which many instances of many applications are executed -- is likely to perform better in a true production environment than a system that does not perform well in this context. A hardware vendor with adequate in-house programming skills can substantially improve application performance and may place these skills at a software vendor′s or customer′s disposal.

6.1 System Performance Evaluation Corporation (SPEC)

SPEC is a non-profit corporation formed to “establish, maintain and endorse a standardized set of relevant benchmarks that can be applied to the newest generation of high-performance computers”. This organization was founded in the belief that the user community could benefit greatly from an objective series of application-oriented tests that can serve as common reference points and can be considered during the evaluation process. While no one benchmark can fully

characterize overall system performance, the results of a variety of realistic benchmarks can give valuable insight into the actual performance of the applications being proposed.

SPEC95 is composed of two suites of benchmarks: SPEC CINT95 (eight computation-intensive integer/non-floating-point benchmarks) and SPEC CFP95 (10 computation-intensive floating-point benchmarks).

The intent of SPEC is twofold:

1. SPEC assembles suites of benchmarks that are generally available in source form. These benchmarks are intended to measure something meaningful and are extensively tested for portability before release. There are strict rules on how these benchmarks must be run and how results must be reported for the trademarked results. SPECint95 and SPECfp95 are the metrics for the current SPEC Release benchmark suite.
2. With the SPEC CINT95 and SPEC CFP95 component-level benchmarks, SPEC now has better and larger benchmark suites to measure the CPU, cache, memory and compiler performance. The benchmarks include SPECint95 and SPECfp95 for speed and SPECint_rate95 and SPECfp_rate95 for computation-intensive throughput. The following categories are part of SPEC95:
   - integer versus floating-point.
   - conservative versus aggressive compilation.
   - speed versus throughput.

Compilation	Speed	Throughput
Aggressive	SPECint95	SPECint_rate95
Aggressive	SPECfp95	SPECfp_rate95
Conservative	SPECint_base95	SPECint_rate_base95
Conservative	SPECfp_base95	SPECfp_rate_base95

Table 17. SPECint95 and SPECfp95 Categories

6.1.1 SPECint92 and SPECfp92

SPECint92 and SPECfp92 are now obsolete and were replaced by SPECint95 and SPECfp95.

SPECint92 — This suite contained six benchmarks performing integer computations, all of them written in C.

SPECfp92 — This suite contained 14 benchmarks performing floating-point computations -- 12 in FORTRAN, two in C.

Technology is always improving. As the technology improves, the benchmarks need to improve as well. Here are some reasons for the SPEC92 withdrawal:

- Modern applications are growing in complexity and size, which made SPEC92 less representative of what runs on current systems. For example, some benchmarks were fitting into cache, which leads to meaningless results.
- Some areas, such as imaging or database performance were not represented.

- The initial hope for a benchmark is that improvements in the benchmark performance will be generally applicable to other situations. As competition develops, however, improvements in the test performance can become specific to that test only. For example, some compilers were able to detect SPEC92 code and to generate a code with strong specific optimization.

Before SPEC92, there was SPEC89, which reported a single figure called a SPECmark.

6.1.2 SPECint95 and SPECfp95

SPECint95 and SPECfp95 were introduced as a replacement for two earlier benchmarks, SPECint92 and SPECfp92.

6.1.2.1 SPECint95

This is a suite of eight computation-intensive (processor, memory system, compiler) integer benchmarks intended to measure "engine horsepower" on applications of realistic size and content. It is important to remember that performance is not just a product of the processor. Memory architecture, cache management and compile capability can also impact results.

Table 18 describes the application areas of each of the benchmarks:

Benchmark	Application Area	Specific Task	Comments
099.go	Game playing; Artificial Intelligence	Plays the game Go against itself.	Cache activity is small, but it does require something larger than 8 MB cache to run well.
124.m88ksim	Simulation	Simulates the Motorola 88100 processor running Dhrystone and a memory test program.	Due to costs of simulation, the choice of binary loaded into the simulator does not have much impact upon the profile of execution, either at the software or hardware level.
126.gcc	Programming and Compilation	Compiles pre-processed source into optimized SPARC assembly code.	The 50 input files mean that this benchmark has the highest number of fork/execs during the benchmark run, and also the highest number of open and other such system calls.
129.compress	Compression	Compresses a large text file (about 16 MB) using adaptive Lempel-Zev coding.	This version has been modified to do its work in memory, rather than reading and writing files.
130.li	Language Interpreter	Lisp interpreter.	Cross-module optimization (inlining) would be helpful in getting best results.
132.ijpeg	Imaging	Performs jpeg image compression with various parameters.	Good opportunities to show off superscaler integer capabilities.
134.perl	Shell Interpreter	Performs test and numeric manipulations.	As much as 10 percent of the time can be spent in routines commonly found in libc.a: malloc, free, memcpy, etc.
147.vortex	Database	Builds and manipulates three interrelated databases.	Has shown itself to be sensitive to how well the system's TLB handler works.

Table 18. Details of SPECint95

For each of the SPECint95 benchmarks, SPEC gives rough estimates of the basic hardware-level activities: cycles per instruction (CPI), I-cache and D-cache activity and TLB activity. This data is only useful for comparing the individual benchmarks of SPEC95.

Metrics and How to Read Them: The results of the individual benchmark test cases are expressed as the ratio of the wall clock time to execute the benchmark compared to a (fixed) "SPEC reference time" (which was chosen as the execution time on a SUN SS/10@40). The results are reported as the geometric mean of the individual ratios or SPECint95.

A separate measure known as SPECint_rate95 has been defined to use these benchmarks to measure the processing capacity of a given system. It measures not only how fast one computation-intensive job can be done but how many tasks can be accomplished for single processors, symmetric multiprocessor systems and RS/6000 SP machines.

6.1.2.2 SPECfp95

This is a suite of 10 computation-intensive (CPU, memory system, compiler) floating-point benchmarks intended to measure "engine horsepower" on applications of realistic size and content.

Table 19 on page 153 describes the application areas of each of the benchmarks:

Benchmark	Application Area	Specific Task	Comments
101.tomcatv	Fluid Dynamics/ Geometric Translation	Generation of a 2-dimensional boundary-fitted coordinate system around general geometric domains.	One of the benchmarks most sensitive to the speed of the memory system. Can be parallelized reasonably well.
102.swim	Weather Prediction	Solves shallow-water equations using finite difference approximations. (The only single-precision benchmark in CFP95).	One of the benchmarks most sensitive to the speed of the memory system. Can be parallelized well.
103.su2cor	Quantum Physics	Masses of elementary particles are computed in the Quark-gluon theory.	Can be parallelized reasonably well.
104.hydro2d	Astrophysics	Hydrodynamical Navier Stokes equations are used to compute galactic jets.	Can be parallelized reasonably well.
107.mgrid	Electromagnetism	Calculation of a 3D potential field.	Should be one of the easiest to parallelize.
110.applu	Fluid Dynamics/Math	Solves matrix system pivoting.	Parabolic/elliptic partial differential equations.
125.tur3d	Simulation	Simulates turbulence in a cubic area.	Has a large 1-dimensional matrix (Fast Fourier Transformation).
141.apsi	Weather Prediction	Calculates statistics on temperature and pollutants in a grid.	Cannot be easily parallelized.
145.fpppp	Chemistry	Performs multielectron derivatives.	Cannot be parallelized (the behavior of each atom is affected by the behavior of each one of the other atoms). Minimal cache footprint. Mostly a test of register allocation/performance. Good exhibition of superscaler performance features.
146.wave	Electromagnetic	Solves Maxwell's equations on a Cartesian mesh.	Double precision.

Table 19. Details of SPECfp95

Metrics and How to Read Them: The results of the individual benchmark test cases are expressed as the ratio of the wall clock time to execute the benchmark compared to a (fixed) "SPEC reference time" (which was chosen as the execution time on a SUN SS/10@40). The results are reported as the geometric mean of the individual ratios or SPECfp95.

A separate measure known as SPECfp_rate95 has been defined to use these benchmarks to measure the processing capacity of a given system. It measures not only how fast one computation-intensive job can be done, but how many tasks can be accomplished for single processors, symmetric multiprocessor systems and SP machines.

6.1.2.3 Usage

SPECint95 and SPECfp95 are most likely to be used to compare systems in technical environments.

A single user running a computation-intensive program might only be interested in SPECint95 or SPECint_base95. On the other hand, a person who has to choose a machine for multiple-running floating-point simulations might be more concerned with SPECfp_rate_base95.

SPECint95 should not be used to size a database server. Use of SPECint95 ratios to upgrade from one database server to another one can lead to meaningless results because SPECint95 uses mainly processor and memory. It does not stress typical features extensively used by databases such as I/O performance, L2 cache miss rate, memory contention, cache coherency, and interprocessor locks.

Having an in-depth understanding of your application can be useful if you want to size a machine with SPECfp95. As we've seen, SPECfp95 is an average. This means that some parts of the benchmarks might have better results than others. For example, the RS/6000 591 has a SPECfp95 of 12.4. If you look in more detail at the results for each part of the benchmark, you can observe that 102.swim results in 33.7 SPEC Ratio and that 141.apsi results in 8.37 SPEC Ratio. So, if your application is similar to 102.swim, you might have better performance than if it is close to 141.apsi. This shows that for one machine, depending on which application you run, the performance might be quite different.

6.1.2.4 Conclusion

SPECint95 and SPECfp95 can be used to compare workstations in technical and scientific environments as long as you have an in-depth knowledge of the user's application and SPEC benchmark. Using SPEC benchmarks without clearly understanding them can lead to meaningless results.

Although pure performance is important, price/performance should also be considered.

6.1.3 SPECweb96

SPECweb96 is a benchmark that focuses on server performance for static Web pages, measuring the ability of the server to service HTTP requests or "gets".

The SPECweb96 workload is based on analyses of server logs from a variety of popular Internet servers and some smaller Web sites. To further validate the workload, data from the analyses was compared to logs from Netscape and CommerceNet. Access patterns to the files were determined according to the analyses of server logs. The access analysis reflects real-world usage for a Web service provider, with certain files being more popular than others.

SPECweb96 is the first step in the quest to provide benchmarks that portray Web server performance as accurately as possible. It is strictly a Web server benchmark. It does not measure Web client, Web client/server, or WAN performance. Future SPECweb releases are expected to address features such as encryption, multimedia, Common Gateway Interface (CGI) and keep-alive performance.

The SPECweb96 workload defines four classes of files to get, based on the following file sizes: less than 1 KB, 1 to 10 KB, 10 to 100 KB, and 100 KB to 1 MB.

There are several files in each class, with sizes distributed evenly through the range for that class. The access patterns to the files were determined from the analysis of the Web server logs. SPECweb96 directs 35 percent of its activity to the smallest class, 50 percent to the 1-to-10-KB class, 14 percent to the 10-to-100-KB class, and one percent to the largest files. Within each class, there are non-linear distributions of accesses, reflecting the fact that certain files are more popular than others. Finally, the number of times each class set is serviced is determined by the amount of throughput the server can handle: A high-end server will service a greater number of files than a small, desktop server.

6.1.3.1 Metrics and How to Read Them

One or more clients are used by SPECweb96 to send the HTTP requests to the Web server. The software then measures the response time for each request. At the end of the benchmark run, SPECweb96 calculates a metric based on the overall throughput, measured as maximum operations per second the server has achieved (operations/s).

6.1.3.2 Usage

SPECWeb96 is most likely to be used to compare Web servers.

LAN characteristic is a very important component of Web server performance.

With each SPECweb96 result, there are “test run details” that show mean response time for each file class. This can be useful if your planned server configuration has a different class file distribution.

6.1.3.3 Conclusion

SPECweb96 is a first step in providing performance comparisons between vendors. Today, it does not reflect all Web server configurations.

Also, it does not attempt to model latency associated with obtaining data across a wide area network (WAN) such as the Internet.

6.1.4 SPEC SFS

The SFS benchmark (also called 097.LADDIS) was established by the SPEC organization in April 1993. The Nhfsstone benchmark existed before the SFS, but it had many drawbacks, the first being its strong dependency on the NFS client kernel. Therefore, no true comparison could be made between the results from different vendors. The SFS benchmark was developed to resolve Nhfsstone shortcomings.

The major SFS improvement is the significantly reduced dependency on client activity. The load on the server side may almost be considered not to be sensitive to the clients’ configurations.

Attention

You should keep in mind that the client CPU speed and its network adapter efficiency affect the measurement, as well as the vendor’s compiler optimization and its RPC library code.

That is why SPEC strongly advises vendors to use high-speed and powerful NFS clients to decrease their influence on results.

The principle of this benchmark is to measure the NFS throughput on the NFS server at several response times (50 ms being the arbitrary response time limit). For now, this benchmark only supports NFS Version 2.

In order to make these measurements, load generators are used to stress the NFS server. These load generators are a series of NFS clients spread among different networks. The idea is to have enough load generators to saturate each network and enough networks to saturate the server at a 50 ms response time level. No requirements are made of the client configurations or of the network, both of which can differ from one another.

The load generators send NFS requests to the server in a controlled way so that the sendings respect the predefined NFS operations mix and file-access policy.

SFS implements the NFS protocol inside the tests. Therefore, the benchmark creates RPC packets directly, and the response time is calculated between the entrance and exit from the user RPC library calls. It means that the client kernel-specific implementations should not interfere with these measures, which is one major goal of SFS.

This benchmark utilizes standard TCP/IP as a means of communication between the various processes running on all the load generators, and UDP/IP for the NFS requests.

The NFS operations mix (the same as Nhfsstone) has been carefully defined to include:

- I/O operations (22 percent *read* operations, 15 percent *write* operations).
- File name and attribute operations (34 percent *lookup*, 13 percent *getattr*).
- The remainder is distributed among six other operations (*readlink*, *readdir*, *create*, *setattr*, *remove* and *statfs*).

The most expensive remote procedure call (RPC) in terms of system overhead is a write. Since all NFS writes are synchronous in nature, an RPC mixture which is heavily weighted with writes will force the client machine to wait for completion of the writes. Typically, an NFS implementation will have a very small percentage of writes as opposed to other operations. It is for this reason that NFS is inappropriate in environments where updates to files are prevalent. Thus, a worst-case scenario would be, for example, sharing a database over NFS. Note that a diskless environment in which the clients are paging also fits this pattern.

Regarding the files, there are no shared files and, therefore, no file access contention.

The network packets are distributed as follows:

- *read* packets: 90 percent of 8 KB data, 10 percent of less.
- *write* packets: 50 percent of 8 KB data, 50 percent of less.

Sequential disk transfer rates will have less impact on NFS performance than the seek ability of a particular disk. Since NFS requests are 8 KB and are most likely of a random nature, the faster the disk can access a particular block the better. Tuning of local disk access on the server should improve the performance of NFS. Also, since reads and, to some extent, writes are cached on both the client and server, the size of real memory will affect performance.

Large directory structures also will adversely effect the performance of NFS. Since a linear search is done on a directory using the lookup RPC, increasing the size of the directory will increase the number of RPCs generated to locate an entry.

Each AIX command generates a characteristic mix of these RPCs. For example:

- `find` implies *lookup*, *readdir*, *getattr* RPCs.
- `cp` implies *read*, *write*, *setattr* RPCs.
- `mv` implies *rename* RPC.
- `ls` implies *lookup*, *getattr*, *readdir*, *readlink* RPCs.

6.1.4.1 Metrics and How to Read Them

A whole SFS test is made of successive LADDIS executions done at increasing NFS workload levels.

So the resulting graph should look like this:

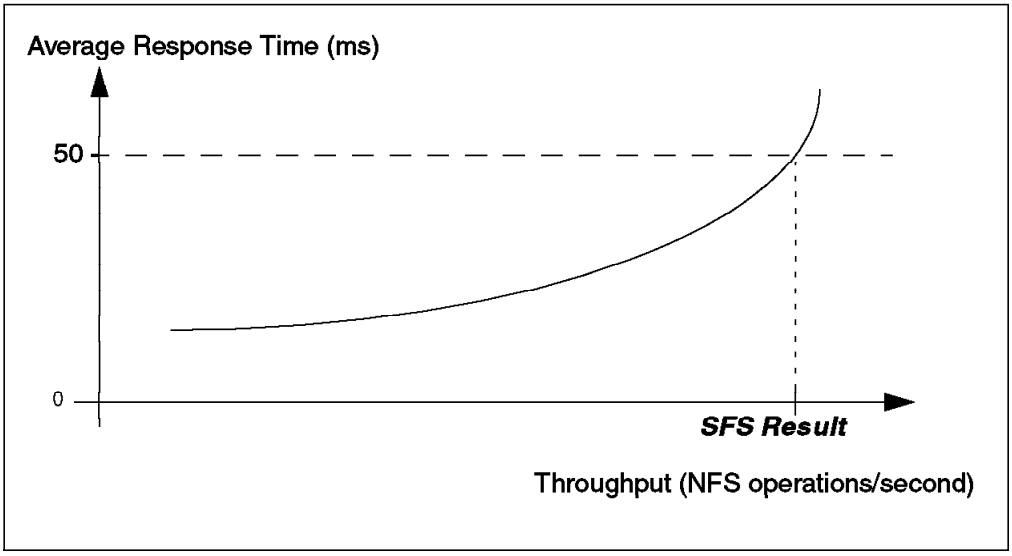


Figure 67. SFS Benchmark Graph

The SFS result is the maximum throughput (in NFSops/s) a non-constrained NFS server can stand while preserving a response time less or equal to 50 ms.

6.1.4.2 Usage

The SFS benchmark objective is to provide a unbiased way of comparing NFS server performances between different systems from various vendors.

What SFS is not

- SFS is neither an NFS client nor an NFS client/server benchmark. It only makes measurements on the NFS server side.
- SFS is not an I/O benchmark, as only a third of the NFS operations mix are read or write.
- SFS does not represent any specific working environment and is only intensively loading the NFS server.

SFS can also be used as an NFS performance tool. Indeed, it is possible to change the default parameters to those of your working environment and then

run a few stress loads to determine the best configuration for your NFS server. But these results do not belong to the SPEC SFS benchmark.

6.1.4.3 Conclusion

SFS will not represent exactly your working environment, particularly your operations mix and data set. Therefore, you should use caution when looking at the results. Indeed, it is extremely difficult, if not impossible in many cases, to predict the performance of a real application using raw benchmark results. But they can be useful in determining the limitations of proposed configurations, and are a means of comparison.

6.1.5 The SPEC System Development Multitasking (SDM) Workload

This benchmark is outdated. We present it for historical reasons and because it is sometimes used for operating system tests.

In 1991, the System Performance Evaluation Corporation introduced a benchmark simulating a multitasking workload by running multiple parallel shell scripts, similar to the AIM suite. This benchmark, called System Development Multitasking (SDM) Workload version 1.0, consisted of two different workloads: 057.sdet and 061.kenbus1, which were run separately. Both parts were meant to simulate an interactive program development environment. Their implementation as shell scripts certainly was a compromise to simplicity. For example, 057.sdet originated from a benchmark that had been used internally at AT&T. It consisted of shell scripts issuing UNIX commands such as `ls`, `cd`, `mv`, `rm`, `echo`, `cat`, `pr`, `mkdir`, `find`, `nroff`, `make`, `sh`, and `cpio`. Likewise, 061.kenbus1 was derived from the MUSBUS benchmark, which was written at Monash University in Australia. It issued UNIX commands such as `cc`, `cat`, `grep`, `rm`, and `mkdir`.

Both benchmarks reported performance in terms of scripts completed per hour. This was done as a peak rate and a curve of throughput vs. workload. Note that, although reporting formats are similar, the 057.sdet and 061.kenbus1 benchmarks were entirely different workloads, and their results were unrelated to each other.

6.1.6 References

On the Web:

`http://www.specbench.org`

Contact:

Standard Performance Evaluation Corporation
10754 Ambassador Drive
Suite 201
Manassas, VA 22110

703/331-0180
703/331-0181 FAX
info@specbench.org

6.2 Graphics Performance Characterization (GPC) Committee

The Graphics Performance Characterization (GPC) Committee has developed into an umbrella organization for autonomous project groups that develop new graphics benchmark methods and performance reporting procedures. In 1996, the GPC joined the SPEC organization.

Current GPC projects are the picture level benchmark (PLB), the X Performance Characterization (XPC), and the OpenGL Performance Characterization (OPC).

6.2.1 PLBwire93 and PLBsurf93

Picture Level Benchmark (PLB) measures the rendering performance of a graphics system through the API. Currently, the PLB benchmark is ported to PHIGS, PEXlib and vendors' proprietary application programming interfaces. PLB was expected for OpenGL by the end of 1996. Performance data from the PLBmark is reported in PLBwire93 and PLBsurf93.

The PLB composites are:

- PLBwire93 is a composite of three benchmarks. These benchmarks are representative of entry-level 3D wireframe applications. These benchmarks are rendered without Z-buffering, depth cueing, anti-aliasing or wide lines.

Name of the Benchmark	Description
sys_chassis	3D wireframe of a computer chassis.
race _car	3D wireframe model of a race car.
seafloor	3D wireframe representing the contour lines of the ocean floor and the islands above.

Table 20. PLBwire93 Benchmark

- PLBsurf93 is a composite of four benchmarks. These benchmarks are representative of applications which shade 3D surfaces. All of these benchmarks utilize Z-buffering and Gouraud shading, and some incorporate multiple light sources.

Name of the Benchmark	Description
cyl_head	3D solid model of an automobile engine's cylinder head.
head	3D human head modeled using data generated by a laser scanner.
shuttle	Low-end simulation consisting of 560 frames.
studio	Architectural walkthrough consisting of 300 frames.

Table 21. PLBsurf93 Benchmark

- The PLB benchmark also reports results from the "oceantopo" benchmark. This benchmark is typical of applications that do mapping of seafloor terrains for exploration purposes.

6.2.1.1 Metrics and How to Read Them

For PLBwire93 and PLBsurf93 benchmark, the result number is given by taking the geometric mean for both PLBlit and PLBopt numbers.

Performance numbers are derived from a normalized constant that is divided by the elapsed time in seconds required to perform the test. Each standard benchmark such as “sys_chassis” has two results:

PLBlit running the benchmark with no optimization for the specific hardware.

PLBopt running the benchmark with an optimization for the specific hardware.

For each composite benchmark, a higher number represents better performance (for example, fewer seconds to display the picture under test).

6.2.1.2 Usage

The PLB benchmark is most likely to compare systems running 3D surface or 3D wireframe applications.

As with any composite, the result is meant to be an overall indicator of performance. Users can look at individual test results to weigh which ones are most important. The composites can be used in the initial analysis of systems. The best test is to run the actual application.

This benchmark tends to mainly stress the graphics adapter. When running a user application, other components are also important for the performance (processor, memory transfer, bus system, L2 cache). This benchmark does not stress these components. This can lead to a real application performance that is not closely related to the benchmark results.

Even if the PLB benchmark stresses mainly the graphics card, for the same graphics adapter, there are some differences in the results with different hardware configurations. With graphics adapter GXT550P, the 43P-140 (166 MHz and L2-512 KB) has a 110.9 PLBwire93 result, and the 43P-140 (200 MHz and L2-1024 KB) has a 123.8 PLBwire93 result.

The PLB benchmark does not require use of the same source code. This means that some source code might be more tuned than others. This makes comparison across different APIs more difficult.

There are a few critical aspects of the user environment that should be understood in order to use the benchmark effectively. Each of the items listed below are often interdependent because computer graphics performance must be approached as a complete system, not as the performance of an individual graphics adapter.

- Application

  It is important to characterize the application requirements (such as dynamic shading) and to quantify the behavior (such as how much rotation).
- API

  Find benchmark results that have been run with the same API, since they will be more representative.
- Main Processor

  In some cases, graphics performance is dependent on the system to which it is attached. In this case, it is important to understand how much CPU power

the application will require. Users should consider this as well as the performance of appropriate benchmarks on varying CPUs.

- Graphics Hardware

  Its performance depends greatly on what is required to do. For example, different APIs may require a graphics adapter to perform a similar task (such as drawing a line) in different ways (usually for compatibility reasons). As a result, unique and often quite different performance results are obtained for similar tasks using different APIs.

6.2.1.3 Conclusion

Benchmark results are a mean value that might not reflect a specific requirement. The PLB benchmark uses static models that are spinning. User applications often use dynamic models that are not represented in this benchmark. Even though the benchmark offers a wide range of 3D utilization, this benchmark does not reflect the entire range of 3D applications. Your application might use specific functionality that doesn't match that used in the benchmark.

Texture-mapping benchmarks were planned for the end of 1996.

6.2.2 Viewperf

The Viewperf benchmark has been proposed by the OpenGL Performance Characterization (OPC) Committee. This benchmark measures 3D rendering performance of systems running OpenGL. One goal of the OPC Committee was to base Viewperf benchmarks on actual OpenGL applications from independent software vendors (ISVs). This was accomplished. The resulting benchmarks are called viewsets. A viewset is a set of Viewperf runs that attempt to characterize the application. Each viewset originates from an ISV. There are currently five viewsets:

Name of the Benchmark	OPC Viewsets	Number of Viewperf Tests	Description
DX-03	IBM's Data Explorer	10	Visualization application.
CDRS-03	Parametric Technology's CDRS	7	Modeling and rendering application for CAD.
DRV-04	Intergraph's Design Review	10	3D computer model review package.
AWadvs-01	Alias/Wavefront's Advanced Visualizer	10	Animation application.
Light-01	Lightscape Technologies' Lightscape Visualization System	4	Radiosity visualization application.

Table 22. Viewperf Benchmark

6.2.2.1 Metrics and How to Read Them

The raw metric for each run of Viewperf is frames/second. (Bigger is better: That is, the larger the results, the faster the performance.) The overall metric for each viewset is the composite, computed using the weighted geometric mean. As with any composite, the result is meant to be an overall indicator of performance.

For each viewset, there is a benchmark result. For example, DX-03 is the result for IBM's Data Explorer Viewperf benchmark; CDRS-03 is the result for Parametric Technology's Viewperf benchmark.

Viewperf measures performance for the following components:

- 3D primitives, including points, lines, line_strip, line_loop, triangles, triangle_strip, triangle_fan, quads and polygons
- Attributes per vertex, per primitive and per frame
- Lighting
- Texture mapping
- Alpha blending
- Fogging
- Anti-aliasing
- Depth buffering

6.2.2.2 Usage

This benchmark is mainly to be used to compare systems using OpenGL in different technical areas: CAD/CAM, image rendering, visualization. Depending on which kind of application you plan to run, you should use the appropriate benchmark.

As with any composite, the result is meant to be an overall indicator of performance. Users can look at individual test results to weigh which ones are most important. The composites can be used for an initial evaluation of systems. The best test is to run the actual application.

This benchmark tends to mainly stress the graphics adapter. When running a user application, other components are also important for the performance (processor, memory transfer, bus system, L2 cache). This benchmark does not stress these components. This can lead to real application performance that is not closely related to the benchmark results.

Even if this OPC benchmark stresses mainly the graphics card, for the same graphics adapter, there are some differences in the results with different hardware configurations. With graphics adapter GXT800P, the 43P-140 (166 MHz and L2-512 KB) has a 17.65 CDRS-03 result, and the 43P-140 (200 MHz and L2-1024 KB) has a 20.49 CDRS-03 result.

All the same restrictions and drawbacks apply as in 6.2.1.2, "Usage" on page 160.

6.2.2.3 Conclusion

A benchmark result is a mean value that might not reflect a specific requirement. An OPC benchmark uses static models that are spinning. User applications often use dynamic models that are not represented in this benchmark. Even though the benchmark offers a wide range of 3D utilization, this benchmark does not reflect the entire range of 3D applications. Your application might use specific functionality that doesn't match that used in the benchmark.

6.2.3 X11perf and Xmark93

The Xmark93 has been developed by the X Performance Characterization (XPC) group for measuring the performance of computers running the X Window System.

- X11perf

  A set of primitive performance tests which measure X Server performance using the Xlib API. The performance results obtained from X11perf are more valuable than hardware primitive rates because X11perf uses the API as an application would use it. The X11perf benchmark suite consists of hundreds of individual primitive performance tests.

- Xmark93

  Because of difficulties in evaluating the hundreds of tests in the X11perf benchmark suite one by one, the X Performance Characterization (XPC) Committee has defined a new performance index called the Xmark93. This is the weighted geometric mean of the 447 X11perf performance tests.

6.2.3.1 Metrics and How to Read Them

By definition, an Xmark93 value of 1.0 corresponds to the X11perf performance of the Sun SPARCstation 1.

6.2.3.2 Usage

The Xmark93 should be only used to measure X11 performance through the Xlib API. It is also important to be familiar with which X primitives the user's application needs and to consult the X11perf information with those primitives in mind.

X11perf is not intended for use as a measure of 3D performance, since Xlib does not support 3D primitives. Xlib is only a 2D API.

6.2.3.3 Conclusion

Even though the benchmark offers a wide range of X11 utilization, this benchmark does not reflect the entire range of X11 applications. The benchmark result is a mean value that does not reflect a specific requirement.

A new X11 benchmark, called XPB, is under development.

6.2.4 References

On the Web:

`http://www.specbench.org`

Contact:

Standard Performance Evaluation Corporation
10754 Ambassador Drive
Suite 201
Manassas, VA 22110

703/331-0180
703/331-0181 FAX
info@specbench.org

6.3 Transaction-Oriented Benchmarks

The Transaction Processing Council (TPC) was founded to define transaction processing and database benchmarks. It also was charged with delivering objective and verifiable performance data to the industry.

TPC is a non-profit corporation of presently more than 40 hardware and software vendors, user organizations and market research companies.

As TPC benchmarks are focused on the overall performance of a system in a transaction-oriented environment, the TPC numbers will be used for comparing computers in a commercial environment.

The actual benchmarks are TPC-C and TPC-D. TPC-A and TPC-B benchmarks are obsolete and therefore only of historical interest.

Attention

Do not compare different TPC benchmarks with each other, as the benchmarks differ completely and do not compare different major versions of the same benchmark.

For additional information, such as the versions of the published benchmarks, check the Transaction Processing Council's homepage at `http://www.tpc.org`.

6.3.1 TPC-A

TPC-A is a very simple, transaction-oriented benchmark in a multiuser environment considering all system aspects including CPU, disk, I/O, memory, terminals, network and operating system.

TPC-A uses a single, simple, update-intensive transaction to load the system as typically found in a simple bank environment. Even though the workload is intended to reflect an on-line transaction processing (OLTP) application, it does not reflect the entire range of OLTP requirements.

TPC-A measures how many transactions per second a system can perform when driven from multiple terminals. The metric for TPC-A is throughput as measured in transactions per second (tpsA).

TPC-A has been withdrawn by the council. Some of the configurations used reflected quite unrealistic conditions, such as tremendous input speeds, unformatted outputs, and only one simultaneous transaction. Nowadays, applications have more complex requirements of a multiuser environment.

TPC-A has been replaced by TPC-C.

6.3.2 TPC-B

In comparison to TPC-A, TPC-B is not an OLTP benchmark. TPC-B is a multitasking benchmark rather than a multiuser benchmark. It is very similar to TPC-A, but it does not include front-end systems. It has been designed for back-end database server stress test.

TPC-B measures how many tasks a system can finish in a specific time. The metric for this benchmark is throughput as measured in transactions per second (tpsB).

TPC-B has been withdrawn by the council. It didn't meet current requirements.

6.3.3 TPC-C

Like TPC-A, TPC Benchmark C (TPC-C) is an on-line transaction processing (OLTP) benchmark. However, it is more complex than TPC-A.

TPC-C is a mixture of read-only and update-intensive transactions that simulate the activities found in complex OLTP application environments.

The company portrayed by the benchmark is a wholesale supplier with a number of geographically distributed sales districts and associated warehouses. The benchmark is centered on the principal activities (transactions) of an order-entry environment involving entering and delivering orders, recording payments, checking the status of orders, and monitoring the level of stock quantities at the warehouses. Each warehouse consists of 10 districts (one terminal represents a district). All five transactions (New-Order, Payment, Order-Status, Delivery, Stock-Level) are available at each terminal. A remote terminal emulator (RTE) is used to emulate the input for an user executing the required mix of transactions over the performance measurement period.

The benchmark has also been defined to meet the circumstances of a realistic environment, as it considers wait times (keying times and think times) experienced by the user. Minimum values for these wait times are defined individually for each of the five transactions.

The database used for the benchmark may be any commercially available database management system (DBMS), database server, file system, or other data repository that provides a functionally equivalent implementation as detailed in the specification of the TPC Benchmark C.

For TPC-C benchmarks on the different RS/6000 models, IBM uses relational databases such as DB2/6000, Informix, Oracle or Sybase. The database used for the TPC-C benchmark can be found by reviewing the executive summary or the Full Disclosure Report from the member companies.

6.3.3.1 Metrics and How to Read Them

The performance metric for the TPC-C benchmark is expressed in throughput as measured in transactions per minute (tpmC).

In TPC-C, throughput is defined as how many new-order transactions per minute a system generates while the system is executing four other transaction types (payment, order-status, delivery, stock-level). All five TPC-C transactions have a certain user response time requirement, with the new-order transaction response time set at five seconds. Therefore, for a 710 tpmC number, a system is generating 710 new-order transactions per minute while fulfilling the rest of the TPC-C transaction mix workload. This means in detail: For every new-order transaction, an equal number of payment transactions (in this example 710), and for every 10 new-order transactions, one delivery transaction, one order-status transaction and one stock-level transaction (in this example 71 of each) will be generated.

The price/performance metric is expressed in price-per-tpmC ($/tpmC).

The cost, which the $/tpmC is based on, is not only the cost of the computer or host machine. It encompasses all of the cost dimensions for an entire system environment the user might purchase. This cost includes communications equipment, software (transaction monitors and database software), operating system, computer systems (server and client), backup storage and maintenance for a five year period. Therefore, if the total system cost is $859,100 and the throughput is 1562 tpmC, the price/performance is derived by taking the price of the entire system ($859,100) divided by the performance (1562 tpmC), which equals $550 per tpmC.

TPC-C is also a very convenient benchmark for symmetric multiprocessing (SMP) systems. The tpmC rates for SMP systems are listed by number of processors.

6.3.3.2 Usage

TPC-C is most likely to be used to compare systems in a commercial environment.

Determining whether the TPC-C benchmark is applicable to a specific application or environment is extremely difficult. The best approach is to reach a deeper understanding of the benchmark and compare its model (user interaction, database design, database size, transaction complexity, processing requirements, storage/backup tests) with the relevant application environment. If there is a rough match, the benchmark data will probably be a useful and relevant tool for comparing different systems that may be installed in the appropriate environment.

6.3.3.3 Conclusion

Even though the benchmark offers a rich environment that emulates many OLTP applications, this benchmark does not reflect the entire range of OLTP requirements.

Benchmark results are highly dependent upon workload, specific application requirements, and system design and implementation. Relative system performance will vary as a result of these and other factors. Therefore, TPC-C should not be used as a substitute for a specific customer application benchmark when critical capacity planning and/or product evaluation decisions are contemplated.

Also, see the introduction to this chapter on page Chapter 6, “Benchmarks” on page 149.

6.3.4 TPC-D

In comparison to TPC-C, the TPC Benchmark D (TPC-D) is not an OLTP benchmark. TPC-D is a decision support benchmark.

OLTP applications are update-intensive and generally consist of shorter transactions that access a small portion of a database, often through a primary key or index. Decision support applications typically consist of long and often read-only queries that access large portions of the database. Database updates are rather subordinate. Thus, the benchmark evaluates the performance of various decision support systems that examine large volumes of data. It executes sets of queries against a standard database with a high degree of

complexity and gives answers to critical business questions. It is a suite of business-oriented queries and concurrent updates executed at any time during the benchmark.

The schema of the benchmark models a database for a worldwide distributor that purchases parts from suppliers and sells them to customers. In order to allow benchmarks on various system sizes, TPC-D provides a scaling factor with database sizes of 1 GB, 10 GB, 30 GB, 100 GB, 300 GB, 1000 GB and 10000 GB (=10 TB). The database size of 1 GB has a scaling factor of 1 (SF = 1). The minimum database required to run the benchmark holds business data from 10000 suppliers. It contains almost 10 million rows representing a raw storage capacity of about 1 GB. Seventeen complex queries have been defined using SQL-92 (using nested table expressions and *case* expressions) plus two update queries that are run against the eight or nine tables of the database. The two largest tables are the master-detail pair of order and line item, which together constitute about 85 percent of the database. The other tables describe the company's parts, suppliers and customers. All tables except the two small Nation and Region *code* tables scale linearly with the size of the database.

A database management system with an SQL interface must be used so that the queries can be run using the syntax provided in the specification. TPC-D chooses to exclusively model ad hoc queries and enforces execution rules which are based upon this premise. Some examples are the restriction on how the queries may be phrased, the ban on optimizer hints, the requirement of a constant DBMS configuration throughout the performance tests, and the requirement of dynamic SQL as opposed to static SQL. For TPC-D benchmarks on the SP, the RS/6000 Division uses DB2/6000 stand-alone. The information on which database the published TPC-D metrics are based may usually be found as additional information to the metrics.

6.3.4.1 Metrics and How to Read Them

There are three different metrics for TPC-D, two performance metrics and one price/performance metric. All of them must specify the database size on which the benchmark was run.

- Query Processing Performance (QppD@size)
- Query Throughput (QthD@size)
- Query Per Hour at Price ($/QphD@size)

Attention

Do not compare metrics of different database sizes!

Query Processing Performance (QppD): This metric provides a measure of raw query execution and shows how fast queries can be executed when all available power of the system is concentrated on a single query at a time. It is also called a *power* metric.

The power metric is based on a geometric mean of the 17 TPC-D queries, the insert test, and the delete test. The inverse of the computed geometric mean is converted to hours from seconds and scaled by the database volume (SF) to obtain QppD.

The formula is:

$$QppD = \frac{1 * 3600}{GM\ (Q1,...,Q17,\ UF1,\ UF2)} * SF$$

GM is the geometric mean. Q is each query, UF are the two update functions.

This leads to the following actual calculation:

$$QppD = \frac{3600}{\sqrt[19]{QI(1,0)*\ QI(2,0)*...*QI(17,0)*UI(1,0)*UI(2.0)}} * SF$$

QI(i,0) is the timing interval, in seconds, of query Qi within the single query stream of the power test. UI(j,0) is the timing interval, in seconds, of update function UFj within the single query stream of the power test.

Query Throughput (QthD): This metric provides a measure of optimal execution of concurrent queries and shows how many queries the system can process in an hour.

It is a classical throughput measure characterizing the ability of the system to support a multiuser workload in a balanced way. A number of query users (S) is chosen, and each executes the full set of queries in a different order. In the background, there is an update stream that runs a series of insert/delete operations (one pair for each query user). The choice of the number of users is at the discretion of the test sponsor.

The throughput metric is computed as the total amount of work (S*17), converted to hours from seconds, scaled by the database volume (SF), and divided by the total elapsed time (Ts) required between the first query starting and the last query or update function completing.

The complete formula is:

$$QthD = \frac{S * 17 * 3600}{Ts} * SF$$

If the test sponsor chooses only one user (S=1) for the throughput test, for ease of benchmarking, it is permissible to omit the throughput test and compute the throughput metric by substituting Ts with the timings obtained during the power metric test. Similarly it is permissible to schedule the insert/delete activity for the throughput test after all the queries have completed.

Query Per Hour (QphD): The primary purpose of the composite Query Per Hour (QphD) rating is to derive a single performance rating that can be used to compute price/performance.

The QphD rating is equal to the geometric mean (square root of the product) of the power and throughput metrics. As a result of combining the power metric

and the throughput metric, QphD places attention on both the ability to make single queries run very fast as well as on the ability to make parallel-executed queries run faster.

The formula is:

$$QphD = \sqrt{QppD * QthD}$$

Price/Performance ($/QphD): The TPC-D price/performance ($/QphD) metric must be computed using the query-per-hour rating:

$$Price\text{-}per\text{-}QphD = \frac{\$}{QphD}$$

In the above formula, $ is the total system price of the tested configuration, including five years of maintenance costs.

Whenever TPC-D results are published, all three metrics (Query Processing Performance@DB-Size, Query Throuhgput@DB-Size and Price/Performance@DB-Size) must be published.

For example:

QppD@size: 835.6@300GB

QthD@size: 364.0@300GB

QphD@size: $32202@300GB

6.3.4.2 Usage

The TPC-D Benchmark will mostly be used to compare decision support systems that are within the range from personal computers at the low-end to symmetric multiprocessing (SMP) and massively parallel processor (MPP) systems at the high end.

6.3.4.3 Conclusion

Despite the fact that this benchmark offers a rich environment representative of many decision support systems, this benchmark does not reflect the entire range of decision support requirements. In addition, the extent to which a customer can achieve the results reported by a vendor is highly dependent on how closely TPC-D approximates the customer application. The relative performance of systems derived from this benchmark does not necessarily hold for other workloads or environments. Extrapolations to any other environment are not recommended.

Benchmark results are highly dependent upon workload, specific application requirements, and systems design and implementation. Relative system performance will vary as a result of these and other factors. Therefore, TPC-D should not be used as a substitute for a specific customer application benchmark

when critical capacity planning and/or product evaluation decisions are contemplated.

6.3.5 References

On the Web:

`http://www.tcp.org`

Contact:

TPC
777 N. First St., Suite 600
San Jose, CA 95112

Fax: 408-295-9768

6.4 OLTP

The Relative OLTP Performance Ratio is an indicator of commercial throughput for RS/6000 servers. It is not a standardized, official benchmark but an IBM model workload to compare RS/6000 servers from an explicit OLTP point of view. Therefore, these ratios are only applicable to RS/6000 servers.

The analytical model represents only kernels of processing found in OLTP systems and reflects only CPU, caches and memory interactions. Disk I/O and network interactions are not directly incorporated. With these limitations, the Relative OLTP Performance Ratio may be used to compare relative performance among RS/6000 servers.

Attention

This ratio does NOT equate to any official TPC-C results from IBM or competition!

The reference system for the Relative OLTP Performance measurement is the RS/6000 Server Model 250, which has a Relative OLTP Performance Ratio of 1.

6.5 LINPACK

The LINPACK benchmark was written in 1983 by Jack J.Dongarra while working at the Argonne National Laboratory. It consists of linear algebra procedures like Gaussian elimination to achieve the LU-factorization of a matrix and solve linear systems of equations using this factorization. The benchmark is written in FORTRAN. It is highly vectorizable for most vectorizing compilers. In the original benchmark source code, the basic linear algebra subroutines, which are called by the higher-level routines, absorb 80 percent of the run-time of the LINPACK benchmark. They are also coded in FORTRAN. As with most benchmarks in the technical and scientific field, both single- and double-precision versions of the source are in circulation. Manufacturers often legitimately modify the program by adapting the matrix size to the cache line length and cache reload algorithm of the system being benchmarked.

The LINPACK 100 x 100 benchmark can be run in single precision (SP) or double precision (DP). No modification of the code is allowed.

The Toward Peak Performance (TPP) LINPACK benchmark was developed to assess the increasing capabilities of hardware and software. In this case, the number of equations is 1000, and optimization of the code is permitted, as long as the required precision of the results is achieved.

To reflect performance of massively parallel processor (MPP) systems like the RS/6000 SP machine, the High Performance Computing (HPC) LINPACK benchmark has been developed. You can scale the matrix size (number of equations) to a value (Nmax) where you can get optimal performance. The size of the matrix (Nmax) and the optimal performance in Gflops (Rmax) are reported. By allowing the matrix order to increase on each system (rather than using a fixed matrix order), the LINPACK HPC benchmark provides an estimate of the highest sustainable application performance. As additional information, you also have to determine the matrix size (N1/2) for half of the Rmax performance that was achieved. Usually, the theoretical peak Mflops result is also given. This number reflects the theoretical maximum performance you could achieve on a particular machine.

6.5.1 Metrics and How to Read Them

The LINPACK benchmarks indicate performance in the unit of millions of floating-point operations per second (MFLOPS), although this is technically not quite accurate. The benchmark calculates the pre-supposed number of operations on the FORTRAN language level from the matrix sizes and from knowledge about the algorithms needed to do the transformations between them. This disregards compiler optimization. LINPACK MFLOPS indicate performance on an application and not a hardware level. Although the unit of measure, MFLOPS, suggests otherwise, for MPPs, values are usually reported in giga of floating-point operations per second (GFLOPS).

6.5.2 Usage

This benchmark does not reflect the overall performance of a given machine, as no single number ever can. It only reflects the performance of a dedicated system solving a dense system of linear equations.

6.5.3 Conclusion

This benchmark is heavily influenced by the size of the workstation's data cache, especially in the case of the two larger matrices. It can be made to fit into the cache by manipulating the matrix size.

LINPACK measures units of work on a problem level masquerading as a hardware performance indicator. It does not take into account compiler optimization like dead code elimination or hardware features like the multiply-add instruction.

6.5.4 Reference

On the Web:

http://www.netlib.org/benchmark/performance.ps

6.6 The AIM Benchmark Suites

AIM Technology is a company that specializes in developing benchmarks, licensing them to computer manufacturers and other corporations, performing measurements with their benchmarks, and selling the result sheets to prospective customers. AIM Technology has a number of benchmark suites available. The most important are the AIM Multiuser Benchmark Suite-VII for performance evaluation of multiuser systems and the AIM Workstation Benchmark Suite-VI for workstations.

6.6.1 AIM Multiuser Benchmark Suite-VII

The AIM Multiuser Benchmark Suite-VII is designed to measure performance of multiuser environments. It is composed of several components that are related to specific workloads: multiuser/shared, computation server, large database, and file server. The benchmark runs each workload and a mix of these workloads.

This benchmark widely uses all resources of a server: CPU, I/O subsystems, memory, and cache.

For each workload, peak jobs/min and sustained load are reported.

6.6.2 AIM Workstation Benchmark Suite-VI

The AIM Workstation Benchmark Suite-VI is designed to measure performance of workstations running multiple workloads in various environments. It includes CAD/CAM, GIS/imaging, general business, software development, and digital design.

This benchmark widely uses all resources of a workstation: CPU, I/O subsystems, memory, cache, compiler optimization, and operating system features.

For each workload, peak jobs/min and sustained load are reported.

6.6.3 References

Contact:

AIM Technology
4699 Old Ironsides Drive
Suite 400
Santa Clara, Ca 95054
USA

Phone: +1-408-748-8649
Fax: +1-408-748-0161
E-Mail: benchinfo@aim.com
WWW: www.aim.com

6.7 Nhfsstone Benchmark

Nhfsstone (pronounced n-f-s-stone, the ″h″ is silent) is a copyrighted product of Legato Systems, Incorporated. This benchmark intends to measure the performance of file servers that follow the NFS protocol.

It is an NFS load generator, based on a load mix coming from studies and experience. An NFS client generates an artificial load with a particular mix of NFS operations. The benchmark reports the average response time of the server in milliseconds per call and the load in calls per second. The benchmark adjusts its calling patterns based on the client's kernel NFS statistics and the elapsed time in order to stay between fixed limits.

Nhfsstone has only a one-client capability.

This benchmark is now obsolete and has been replaced by the SFS benchmark.

6.8 WebStone

WebStone is a benchmark developed by Silicon Graphics that focuses on performance for Web servers. It measures the ability of the server to service HTTP requests or "gets".

The WebStone workload is based on analyses of server logs from a variety of popular Internet servers and some smaller Web sites. To further validate the workload, data was gathered from different sites:

Hotwired (http://www.hotwired.com)
IUMA (http:/www.iuma.com)
Netscape Communication (http://www.netscape.com)
Silicon Graphics (http:www.sgi.com)

The WebStone 1.1 workload defines four "mixes" that try to represent a general use of HTTP:

Name	Description	File size
General modem mix	Synthetic mix of pages that would be accessed by modem users.	Less than 20 KB.
General mix	More concerned with the network responsiveness and throughput.	Between 1 KB and 100 KB.
Media-rich mix	Uses multimedia content such as: MPEG and Quicktime movie files, sound clips.	From 20 KB to megabytes.
General and media-rich mix	Simulates site that has both small content and media-rich content.	From 1 KB to megabytes.

Table 23. WebStone 1.1 Mixes

6.8.1 Metrics and How to Read Them

WebStone reports throughput (Mb/s), connection rate, and latency (time to complete a request) related to the number of clients. WebStone also measures error rate and Little's Load Factor. Little's Load Factor is based on queuing theory and is a ratio of time spent talking to the Web server versus time waiting for the server to respond. In a WebStone test, Little's Load Factor should be just under the number of WebStone client processes -- higher is better. A typical report would be:

Client	Connections/s	Errors/s	Latency	Little's Load Factor	Throughput (Mb/s)
128	139.65	0.000	0.9075	126.73	7.61

Table 24. Example of WebStone Results

6.8.2 Usage

WebStone is most likely to be used to compare hardware and software Web servers.

6.8.3 Conclusion

WebStone is a first step in providing performance comparisons between vendors. LAN characteristics are very important in Web server performance.

WebStone 2.0 is just now available. It includes new functionality like proxy servers, Common Gateway Interface (CGI) and Netscape API (NSAPI) programs. Some rules have been changed to make benchmark results more meaningful.

WebStone 1.1 and WebStone 2.0 results should not be compared, as the benchmarks are different.

6.8.4 References

On the Web:

`http://www.sgi.com/Products/WebFORCE/WebStone`

6.9 NotesBench Benchmark

NotesBench is the Lotus-authorized performance benchmark suite, measuring up to six workloads by emulating the traffic that LAN-attached clients would generate when executing these workloads.

It is a tool developed by Lotus that enables hardware vendors to directly provide Lotus Notes customers with Notes capacity and performance information on various platforms and configurations.

NotesBench is only available to hardware vendors and Lotus Premium Business Partners who have fulfilled the prerequisite three-day hands-on NotesBench training, not to customers.

The NotesBench software consists of a suite of benchmarks. Each benchmark maps to a workload or test, and each workload models Notes workstations-to-server or server-to-server operations. The six NotesBench tests

are described below. Please use these terms only to describe NotesBench workloads.

Idle Usage Test: This workload establishes an upper bound on the number of sessions that a Notes server can support. After establishing sessions between client and server, the test carries out no Notes transactions in the sessions. Other than the resources required to start a session, no other resources are used in the test. The resultant capacity metric is the maximum number of user sessions that can concurrently exist.

Replication Hub Test: A replication hub is a Notes server that exists to propagate changes in Notes databases among a collection of other servers. The workload for a replication hub consists of replicating changes to user databases. Typical replication hubs also have some amount of replication load for the Name and Address Book (NAB) database, but that is not included in the NotesBench replication hub workload. The results for the replication hub test are the throughput metric (the number of documents replicated per unit of time), the average response time, and the number of spoke servers supported. The test procedure for a replication hub uses a hub-and-spoke topology. The number of driver systems serve as source and destination spokes, and the system under test serves as a single replication hub. The spoke servers modify local replica databases and then replicate those changes to the hub server. Each test script user makes changes to one local database and then replicates those changes to the server. The server runs replicator and updater but no other server programs. The test executes these database modifications on the spoke systems: additions, updates, categorizations, and deletions.

Mail Routing Hub Test: A mail routing hub is a server that exists to route messages to other servers (a ″pure″ router) and possibly to deliver messages to local users. The workload for a mail routing hub, the system under test, consists of receiving messages from source driver systems and routing or delivering each message to a destination system. The test results for the mail routing hub are the throughput metric (the number of messages processed, routed or delivered, per unit of time, messages transferred to recipients per hour, and message bytes transferred to recipients per hour).

Mail Test: This workload executes Notes transactions that model a server for mail users at sites that rely only on mail for communication. The resultant capacity metric for a mail-only server is the maximum number of users that can be supported before the average user response time becomes unacceptable. The results for the mail test are throughput of completed Notes operations, average response time at maximum capacity, and maximum mail users supported.

About every 15 minutes, each simulated user will perform the following:

- 5 documents read
- 2 documents updated
- 2 documents deleted
- 1 view scrolling operation
- 1 database opened and closed
- 1 view opened and closed
- Some miscellaneous operations

Every sixth iteration through this script, a 1000-byte mail message is sent to three recipients.

Mail and Shared Database Test (MailDB): This workload models a server for active users who are only performing mail and simple shared database operations. The test includes mail-only activity plus view operations and navigation of unread documents in a shared database. It applies especially to sites that rely primarily on mail for communication or that have Notes users who do not yet use all Notes features. The results for the Mail and Shared Database test (MailDB) are throughput of completed Notes operations, maximum users supported, and average response time at maximum capacity. The throughput for this test is a capacity metric. It is the maximum number of active users that can be supported before the average user response time becomes unacceptable.

About every 15 minutes, each simulated user will perform the following:

- 8 documents read
- 2 documents updated
- 2 documents deleted
- 4 view scrolling operations of 20 rows each
- 2 databases opened and closed
- 2 views opened and closed
- Some miscellaneous operations

Every sixth iteration through this script, a 1000-byte mail message is sent to three recipients.

Groupware A (Complex Mail and Interactive User Test): This workload models a server for experienced Notes users who are sending large mail messages, adding documents with attachments to shared databases, performing full-text searches, and replicating changes from their local machine to the server. The Groupware A test includes mail and shared database activity plus mail messages with 300 KB body fields, mail messages with 500 KB attachments, users that replicate with the system under test, and users that execute full-text searches of a shared discussion database. The results for the Groupware A test are throughput of completed Notes operations (NotesMark), maximum users supported, and average response time at maximum capacity. Groupware A is a capacity test for Notes users that process large amounts of information. This workload models sites that use the most resource-intensive features of Notes. It can be used to establish a worst-case lower boundary on the maximum number of users a server can support. The resultant capacity metric for a power-user server is the maximum number of users that can be supported before the average user response time becomes unacceptable.

The script for this benchmark includes all the MailDB activity, plus the following:

- each scrolling operation is for 40 rather than 20 rows
- one mail message has a large body (32 KB)
- second mail message has a large attachment (500 KB)
- changes are made to a local database and pushed to the server
- full-text search of a discussion database

6.9.1 Metrics and How to Read Them

NotesBench generates the same throughput metric for each of its workloads (the value of the metric changes from test to test). This metric is called a NotesMark and has the units of transactions per minute (tpm).

Along with a NotesMark value, each workload produces a value for the maximum users supported in the test and for the average response time.

Thus, an audited report would look like this:

- Platform description (detailed hardware and operating system configurations)
- Results
- Analysis
- All the Notes parameter optimizations

Attention

Be aware that some vendors may be using their own, non-NotesBench workloads to define Notes server performance and that they have been using terminology similar to NotesBench.

For example, one vendor is claiming a very high number of ″power″ users. The claims are not NotesBench workloads or audited NotesBench results. This ″power″ user sends mail messages as follows: 50 percent 1 K messages, 20 percent 5 K messages, and 30 percent 10 K messages. The NotesBench ″power″ user test (Groupware A) consists of 100 percent 800 K mail messages!

Be on the lookout for non-audited numbers.

6.9.2 Usage

Notes benchmarking is designed to answer the question of ″which server configuration is better.″

This enables an apples-to-apples comparison for a particular Notes version (e.g. AIX with 256 MB RAM vs. AIX with 512 MB RAM) or across different server versions.

Important

NotesBench has been a volatile program since its inception. NotesBench can
change with each build of the Notes product. The workloads can change, and historically, they have ″lightened″, meaning later tests can produce better results than earlier tests.

You must always know the release (and even the build number for pre-audited results) of Notes/NotesBench used for particular runs.

The various NotesBench workloads have been designed to emulate different user behavior, from new Notes users (Mail), to moderate users (MailDB), to power users (Groupware A).

However, it would be inappropriate and inaccurate to use these as exact capacity and performance results within a production setting. Technology factors (including CPU types, OS versions, peripherals, etc.) change too often to give a valid long-term answer. Therefore, the published results should be used primarily as baselines (from which adjustments are made up or down) for a customer′s unique environment:

- Adjust the user count up or down by some ratio, based on customer's end user usage behavior.
- Use published ″throughput″ data (such as the replicator hub workload) to determine if the bytes transferred mirror the customer′s replication behavior.
- Use the hardware configurations documented in the benchmark reports as guides for selecting and optimizing hardware on proposed systems.
- Monitor the changes made to default settings (such as operating system settings and Notes settings) to determine optimization settings for specific workload levels.
- Monitor network topology setups documented in tests to determine if unique components are being used for testing (such as PCI network interface cards (NICs), dual NICs, etc.).

Furthermore, Lotus Consulting is able to provide professional assistance using NotesBench and other tools within a customer′s environment.

6.9.3 Conclusion

The Groupware A workload seems to be the most realistic workload for large organizations implementing Lotus Notes, since it deals with both intensive mail and shared databases. Otherwise, depending on the estimated Notes usage, you should refer to the closest NotesBench workload.

6.9.4 References

Further details or numbers may be found at the followings URLs:

`http://www.lotus.com/`	Lotus home page
`http://www.lotus.com/ntsdoc96/`	NotesBench results
`http://www.support.lotus.com/`	Lotus support home page
`ftp://ftp.support.lotus.com/`	Lotus ftp

Chapter 7. Sizing

Any sizing task implies complex and error-prone procedures. There are many factors making a sizing job inadequate or making it deviate from the final target. A few of the major factors are listed below:

- **The speed at which technologies change continues to accelerate**

  Technology evolves more and more quickly. Nonetheless, the best decision is not always to postpone purchases until the exact adequate product is available. In almost all cases, configuring the latest and state-of-the-art technology works best, keeping in mind that the price/performance ratio plays an important role when choosing between models and capacity features of computers.

- **Every customer has his particular requirements**

  Following system sizing and installation models that others have made is not an easy job. Many organizations have common features at the moment they choose a certain technology, but there are many different factors that differentiate their sizing models:

  - Quantity and distribution of the application users.
  - Required response times for end users and the administrative staff.
  - Security, availability and integrity levels for both applications and data.
  - Application nature, data distribution and sharing models.
  - User-authorized use of system resources (such as applications, personal storage space, printer spoolers, mail, backup tools).
  - Types of concurrent applications running on the same server.
  - Quantity and distribution of data in the system.
  - Type and version of enablers (like DBMS) and applications.

- **Data for sizing is often insufficient**

  It is usually hard to find enough data concerning the resource consumption of an application (like processor, memory or I/O). And even if you had this kind of information, there is no accurate model that is able to identify the most appropriate RS/6000 system(s) other than simulation.

- **New data processing needs may arise**

  Plans for a data processing system normally are performed following established requirements. However, new needs may appear during the installation or production phases. It is then necessary to provide additional resources.

- **Predicting the future is not easy**

  The organization's strategic plans, those that form the basis of technological projections, are affected by internal and market-driven decisions. Those modifications imply in many cases that a well-planned project or one in the developing stage becomes obsolete before completion. The sizing can also reveal itself to be underevaluated or the architecture unscalable.

Last, but not least, the goal is to satisfy all the customer's expectations and needs in the most reasonable way. In a well-sized system, the configured equipment is used adequately, and additional components are configured to support peak levels or marginal growth.

7.1 General Sizing Concepts

The goal of this part is to give a few general guidelines relevant for almost any type of sizing. We then will review some of the most important concepts influencing the sizing procedure.

7.1.1 Guidelines

Be sure to choose your system(s) carefully so that you have some scalability and/or expansion capacities left. This should include:

- Processor

  Must not be busy all the time, or should have some upgrade available.
- Memory

  Should not be configured near its upper limit.
- Number of slots

  Try to have a few slots available for later evolutions (such as disk drives).
- LAN

  Configure enough LAN adapters so that this does not become a bottleneck.

  Communication is often forgotten in sizing procedures, but it is a major issue, especially as LAN bandwidth is strained by more-demanding client use and applications. Like users or applications, communication demands special performance tools and capacity planning. LAN hardware and related issues are documented in 4.5, “LAN / WAN Adapters” on page 90.

  Whenever possible, try to split the different services available on the network to several servers.

It is generally known that the more complex the system, the more likely it is to get stalled by the weakest component that is not properly analyzed and configured.

As a general rule, real workload simulations are obviously more accurate than rules-of-thumb sizing or even official benchmarks, since the latter do not represent a genuine customer application.

Keeping statistics for actual resource consumption and request occurrence will help when configuring updates or expansions to existing systems. There are some tools that help in the sizing process, like BEST/1 from BGS Inc. (Refer to 8.4, “BEST/1” on page 284 for more details.) Those tools are, in many cases, the only ways to model and understand workload tendencies in large, heterogeneous or complex systems.

A complete model for hardware resource sizing must consider at least the following three factors:

- server(s) sizing
- client workstations or personal computer sizing
- communication adapters and bandwidth sizing

Attention

Keep in mind that the load generated on an RS/6000 system does not only come from application-related tasks but also from administrative and/or system-management subsytems.

For instance, a large backup can require an important part of the LAN bandwidth.

7.1.2 Concepts

Workload: Workload can have different meanings, depending on the context. In benchmarks, workload is defined as a set of test executions of a particular application. Here, the term workload is used as the load a system has to bear due to an application.

The nature of the typical workload must be known before any sizing plan can be implemented. Workloads can be categorized as follows:

- **Interactive**

  The user has a live session in the system, using a text-based or graphical terminal or a client computer to control his work.

- **Background**

  This kind of workload requires the users to be logged into the system until the job ends, but it does not require any interaction with the user. A background workload is usually made of processes and subsystems servicing interactive user processes, like printer spool daemons, TCP/IP or SNA daemons, or database server processes.

- **Batch**

  It is basically like the background workload, but the user does not have to be logged into the system to allow batch processes to run. These kinds of jobs do not need to interact with the user for input or output. The usual work done in batch is database file updates, reports created in text files, packets sent over the net, and so on.

A good recommendation is to avoid, as much as possible, mixtures of different heterogeneous types of workloads. This should be done in order to help the performance-management and capacity-planning tasks, and also because of possible resource contention.

Workload distribution throughout the day should be studied, as shown in Figure 68 on page 182, to be able to estimate the peak workloads.

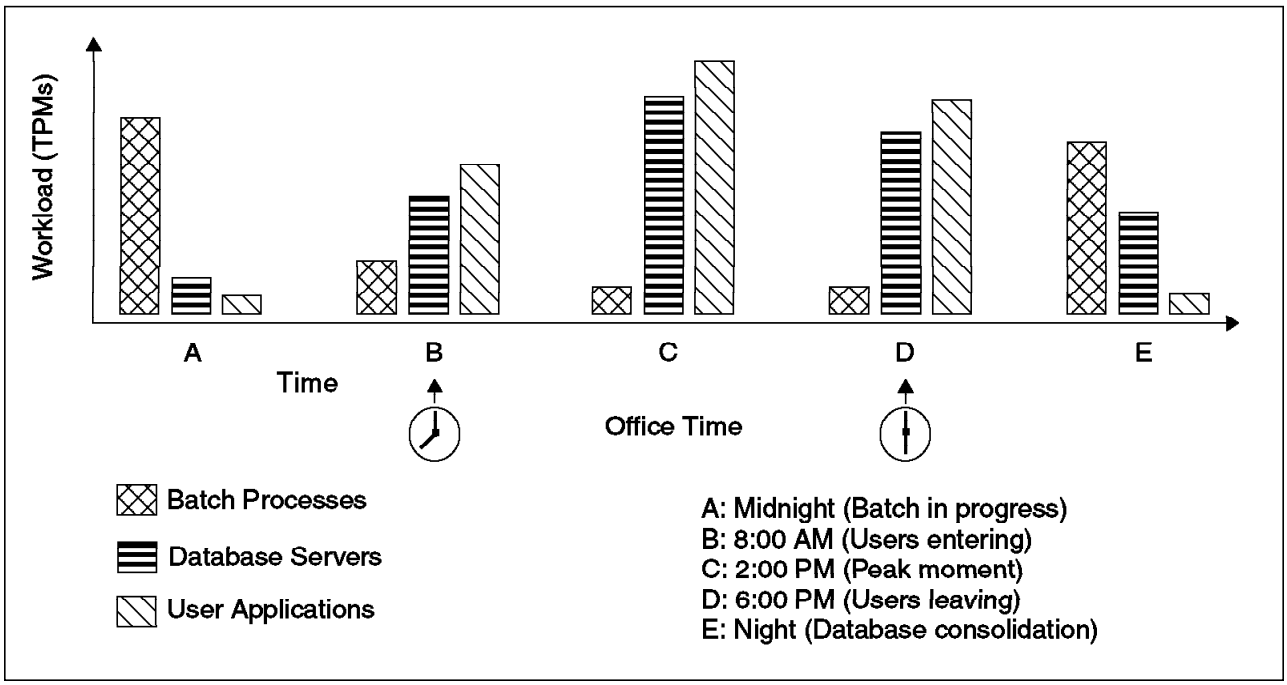


Figure 68. Example of Heterogeneous Workload Type in a System

The following formula gives a rough idea of how to evaluate your application workload.

MWD = Max[**W(Morning), W(Noon), W(Afternoon), W(Night)**]

For mission-critical applications:

Workload = Maximum[**MWD(Monday), MWD(Tuesday), MWD(Wednesday), MWD(Thursday), MWD(Friday), MWD(Saturday), MWD(Sunday)**]

For other applications:

You may want to size for a lower workload, which should be at least the average value of all the maximums, and preferably closer to the maximum.

Workload = Average[**MWD(Monday), MWD(Tuesday), MWD(Wednesday), MWD(Thursday), MWD(Friday), MWD(Saturday), MWD(Sunday)**]

Where: W(n) = Workload at moment n.
MWD(d) = Maximum workload in the day (d).

Figure 69. Workload Formula

A system should be configured with spare capacity to support both the typical workload and peak levels with acceptable response times.

Applications: One of the most important tasks to do before sizing RS/6000 systems as servers or workstations is to know what kind of applications will run on the system.

The applications may be categorized depending on their program construction and interaction as follows:

- Monolithic
- Modular
- Intensive computing
- Interactive
- Batch
- Clients of a local or remote server
- Servers of local or remote clients

Monolithic applications are able to run by themselves. They need no external interaction (for example, complete application programs written and compiled in C or Cobol).

Modular applications are those that group a set of specific programs loaded one at a time. Each program belongs to a complete module, and a module is in charge of a specific end-user task. Such applications include data entry and report modules operating over the same data but separated in different programs and accessed through a menu.

Applications can be of more than one type at a time. For instance, an intensive computing application may be a monolithic program.

Figure 70 gives an idea on the resource requirements of each application type.

APPLICATION TYPE	PROCESSOR Demand	MEMORY Demand	I/O Demand	LAN TRAFFIC	MAX NUMBER IN THE SYSTEM
Monolithic	Application Dependent	**HIGH**	Application Dependent	Application Dependent	Application Dependent
Modular	Application Dependent	**LOW/MEDIUM**	Application Dependent	Application Dependent	Application Dependent
Intensive Computing	**HIGH**	Application Dependent	**LOW**	**LOW**	**LOW**
Interactive	**LOW/MEDIUM**	Application Dependent	**LOW/MEDIUM**	**LOW/MEDIUM**	**MEDIUM/HIGH**
Batch	**MEDIUM/HIGH**	Application Dependent	**MEDIUM/HIGH**	Application Dependent	**LOW**
Client	**LOW/MEDIUM**	**LOW/MEDIUM**	**LOW**	**LOW/MEDIUM**	**MEDIUM/HIGH**
Server	**MEDIUM/HIGH**	**MEDIUM/HIGH**	**HIGH**	**MEDIUM/HIGH**	**LOW**

Figure 70. Application Types and System Loads

Mixing different kinds of applications together on the same system can result in contention (keeping some resources out of the reach of other ones); this should be avoided. Furthermore, hardware specialization of servers (like file servers, database servers, and so forth) has the advantage of making sizing and system management easier to perform.

Concurrent User: It is very important for sizing to understand the difference between a user that is authorized to log on to a system, a connected user and a concurrent user who is actually logged in and using the application. When sizing the peak workload, you should focus on the maximum number of concurrent users, not the total number of users. When sizing disk and memory capacities, you should focus on the maximum number of connected users.

When speaking of concurrent users, you have to assume they have the same working habits so that they form a homogeneous population generating a steady workload. Otherwise, there is no way to model their load generation.

Response Time: The response time question has two parts:

1. How long should the application response time for a typical user of profile X be so that his work is not affected by the system?
2. How much will it cost to decrease the response time for user X to the expected response time Y from Z?

For mission-critical applications, the first question is the most important. There is nothing to add about performance/price considerations.

For conventional applications, you should focus on the second point.

Queuing Concept: Internal physical resources of a computer are managed with request queues (processor, I/O adapters, physical disk drives). The fact that the queues have elements inside is not a problem. The problem arises when slow resources have many requests and the rate at which elements are queued becomes greater than the rate at which the requests are being dispatched. When this happens, the queue grows following an exponential model.

Response time soars every time the system is used at 100 percent, and it could be necessary to reduce the active jobs to allow the system to come back to a steady state (see Figure 71).

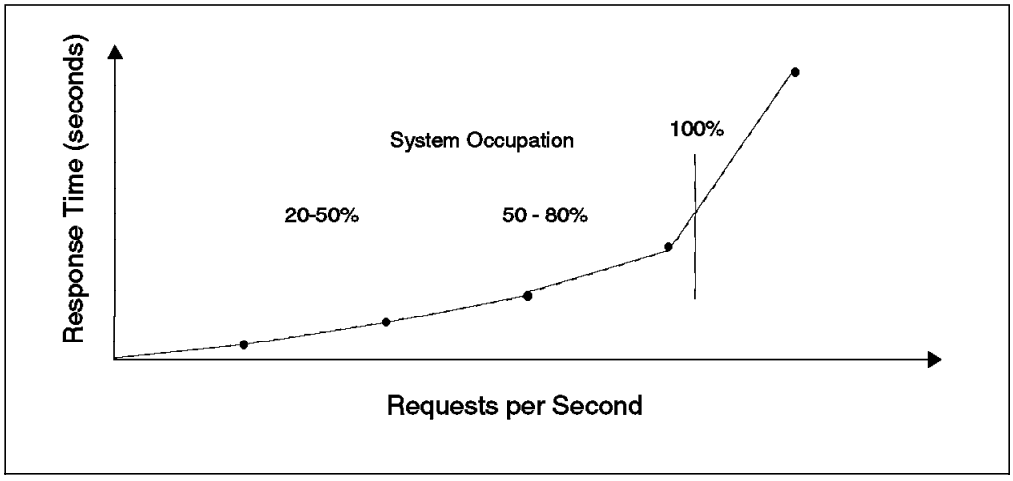


Figure 71. Response Times and System Occupation

As a general rule, you should not configure a system to use more than 75 percent of its processor capacity, or 40 percent of its disk throughput abilities. This provides surplus capacity for peak moments.

Component Speed-Up: Overall application performance relies not only on processor speed but also on every component of the system architecture (like memory, disk drives or buses).

The global time an application needs to be executed is obviously the sum of all of the time spent in each computer component.

```
Effective_Application_Time = Sum( Processor time, Memory time, Bus time
                                  I/O time, Disk Drive time, LAN time )
```

Figure 72. Effective Application Time

Increasing the speed of one component may have no influence whatsoever on application performance. It depends on the dependence level the application has with the component.

Let us refer to each of the *Effective_Application_Time* component fractions by the following abbreviations:

PR = Processor time fraction
ME = Memory time fraction
BU = Bus time fraction
IO = I/O time fraction
DD = Disk Drive time fraction
LA = LAN time fraction

The sum of these fractions is 1 (like percentages).

Now, for example, if we increase the LAN speed by a factor of K, the time the application will spend in the LAN component will decrease by a factor of (1/K). This implies that the total application time will decrease by a factor of (1/S), S being the application speed-up (K>S>1). Figure 73 shows this law, known as Amdahl's Law for speed-up. (Another example of this law is demonstrated in 4.3.1, "Performance View" on page 60.)

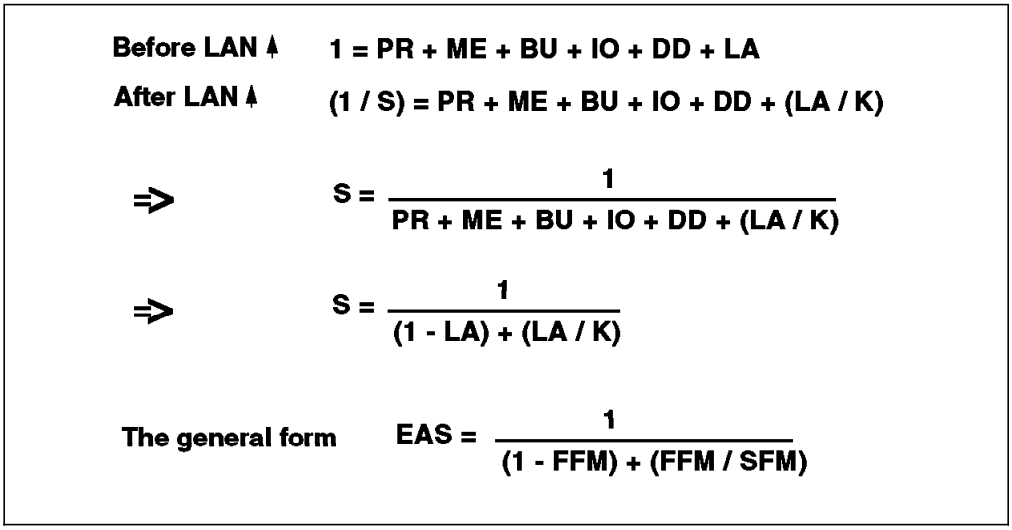


Figure 73. Example of Amdahl's Law

Where:

EAS Effective application speed-up
FFM Fraction of the time the application spends in the improved component
SFM Speed-up of the improved component

Consider two applications in the same system, the first CPU-bound and the second I/O-bound. Consider the fraction the applications are in CPU as 80 percent and 20 percent, respectively. If the CPU is improved by a factor of 2, the speed-up values for both applications are 1.67 and 1.11, respectively.

Processor Speed-Up: An RS/6000 system model typically differentiates itself from its predecessor with its improved processor speed, cache speed, cache size, and bus width. You should not compare two different systems only using

their processor speed; you also need to consider disk, I/O and communication subsystem performance.

Figure 74 shows how a workload derived from a lab-tested end-user application performs in different RS/6000 systems belonging to the same family.

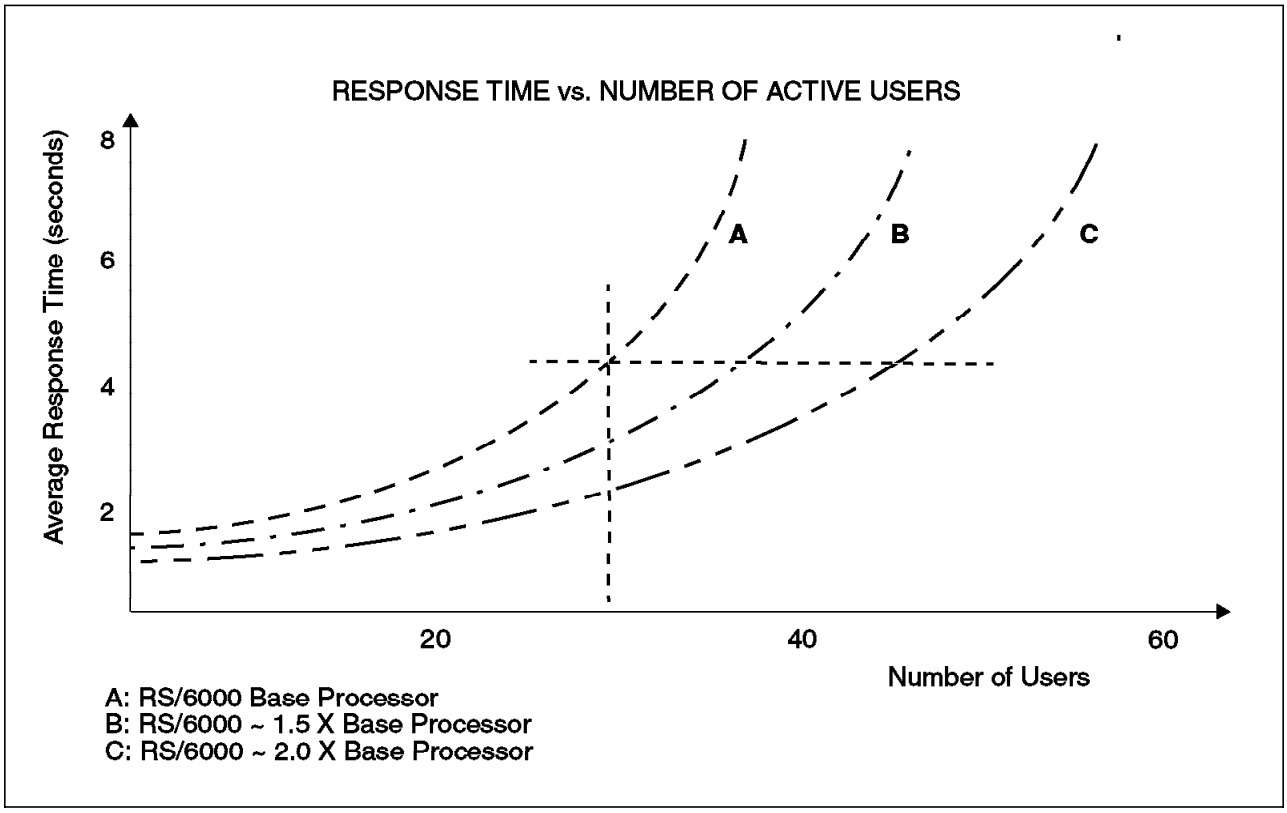


Figure 74. Response Time vs. Number of Active Users

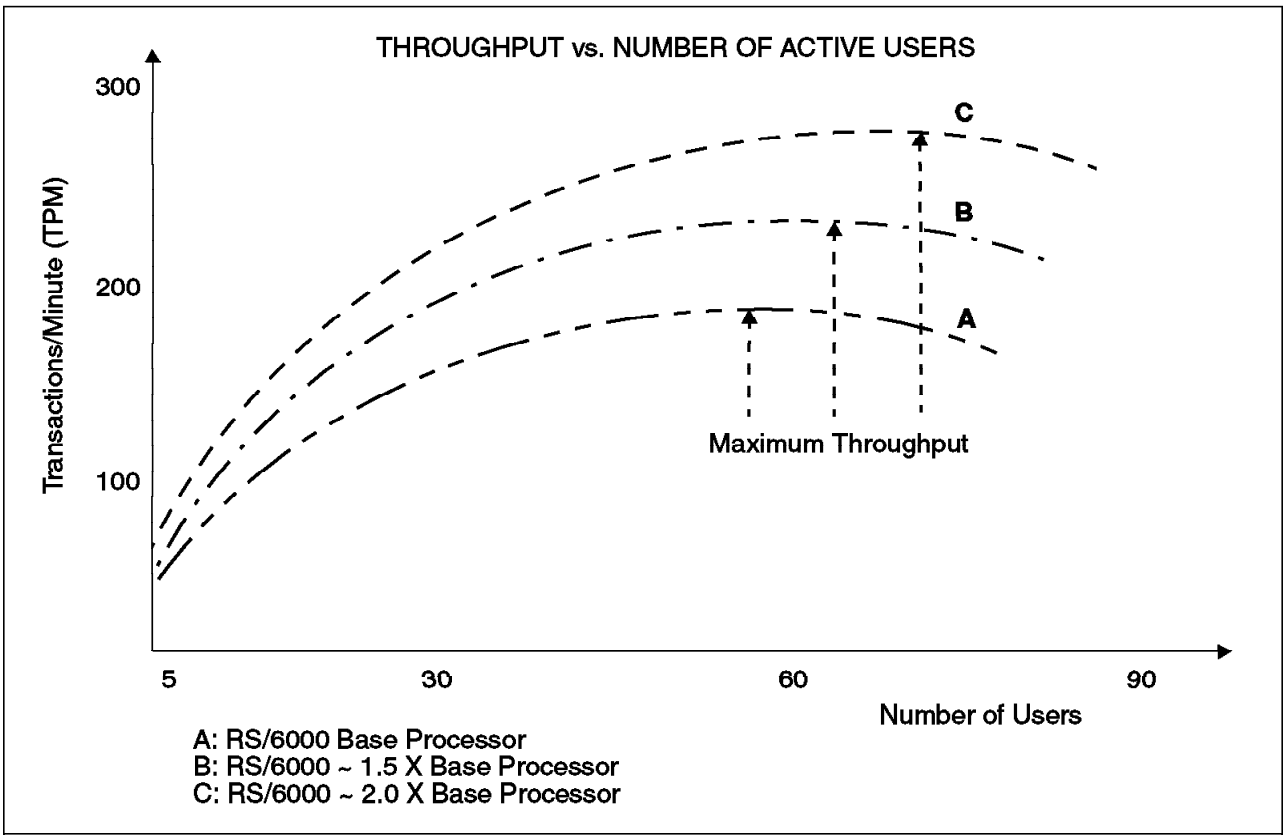


Figure 75. Throughput vs. Number of Active Users

Figure 75 shows the throughput point of view for the same example. Note a special situation: when the maximum throughput is reached in these systems, adding more users results in response-time degradation due to lock contention.

7.2 Workstation Sizing

A common use of workstations is to perform scientific and technical jobs in a highly reliable, graphics environment. RS/6000 workstations are generally used with technical applications such as Geographic Information System (GIS), animation, computer aided design (CAD), computer aided manufacturing (CAM), computer integrated manufacturing (CIM) or mathematical analysis. But they also are excellent for other applications with high-resolution graphics, including Tivoli TME10, multimedia, desktop publishing, and financial applications.

Considerations for configuring workstations involve three main issues:

- Graphics environment
- Processor
- Memory

Although applications running on workstations demand good performance from the I/O subsystem, this is not a critical factor. The reason for this assumption is that most of the applications will load frequently used data into memory, thus minimizing disk access.

7.2.1 Workstation Environment

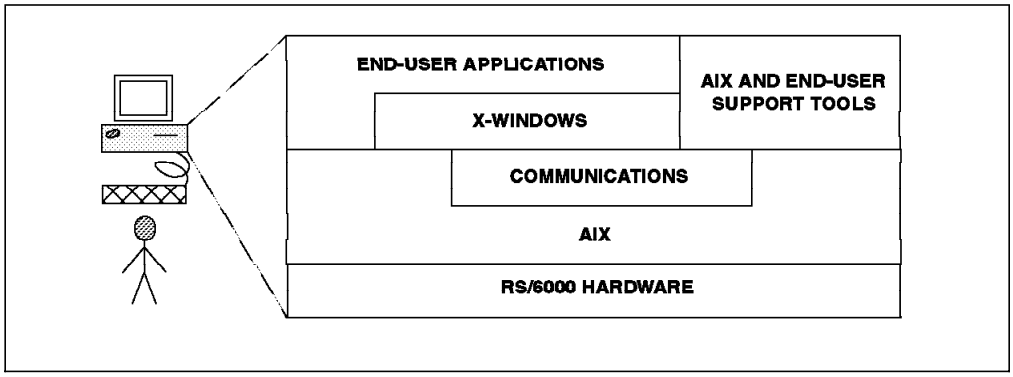


Figure 76. Workstation Software Architecture

No matter what the intended use of a stand-alone system is, it is very common to find that applications on workstations are actually clients of other applications located on other systems in the network (NFS or DFS file servers, for instance). These applications must be taken into account when sizing a new workstation.

Since graphics performance is usually the most important factor, the target system selection should be driven by a combination of processor(s) and display hardware, instead of LAN or disk I/O.

Considerations about asynchronous and/or high-performance LAN adapters may apply in some environments, such as those running multimedia or near-real-time applications. For those cases, you might refer to 4.4, “Asynchronous Communication Adapters” on page 86 and 4.5, “LAN / WAN Adapters” on page 90.

Let us examine more deeply the various workloads that are encountered on a workstation.

Graphics Workload: The graphics display workload comprises all the workload supported by the graphics subsystem in relation to formatting, processing and displaying application graphics data on the user screen. The graphics subsystem is composed of both hardware and software. This implies that graphics performance scalability depends not only on the host processor but also on the graphics hardware and software. The software part is a combination of graphics APIs, X Windows, Motif and the application itself.

For more detailed information, please refer to 4.6, “Graphics Adapters” on page 99.

Processing Workload: As explained in 4.6, “Graphics Adapters” on page 99, the workload a processor has to handle consists of the traditional computing operations plus, depending on the adapter class, the display function executions (which demand high floating-point performance).

Software running on workstations has various requirements. For instance, Internet navigators (browsers) demand good integer performance from the workstation but no 3D capabilities from the display subsystem; GIS applications demand good floating-point and 3D performance from the system. However, both of these graphics presentations look the same on the screen.

As the number of heavy floating-point operations in a graphics environment may be high, it is sometimes necessary to provide more than a powerful graphics adapter to reach the appropriate performance. Many applications (especially scientific applications) require RS/6000 systems with excellent floating-point capabilities, like POWER2- or P2SC-based systems.

I/O Workload: Most applications designed for stand-alone systems handle their own data. But it is common to find applications using a local DBMS manager or being a client of a large database server in a network.

There can be three implementations of the workstation I/O subsystem:

- Local disk storage
- Client of a server in another system
- Client of a file server

Almost all of the I/O activity in a workstation is initiated by a set of few applications. This means the locality of reference for data files is better than in multiuser systems.

The problem in a workstation is not the disk storage technology but the file distribution. Workstations now provide SCSI-2 F/W adapters and disk drives with capacities ranging from 1 GB to more than 10 GB.

The major issues when configuring disk drives for workstations are:

- Sufficient disk storage space to allocate paging spaces and still have some space left for future growth.
- Enough disk drives to perform a good file distribution (separation of transaction files from other storage).
- Configuring duplicate disk drives to improve data availability by mirroring (when applicable).

If your application demands particularly high performance from the disk storage subsystem, then you should refer to 4.3, “Storage” on page 60, to configure the appropriate disk storage subsystem.

Response Time: The behavior of a stand-alone system supporting a specific workload is more predictable than that of a multiuser system because there is only one person involved and a few applications running.

The response time in a workstation is seldom a problem, but some users may need particularly fast response times. In those cases, look for stand-alone systems giving the fastest response times when you really need them.

Many software vendors describe the hardware requirements of their products using standard benchmark numbers. Therefore, if you know how much your application requires in terms of benchmark results, then it will be easier to configure the right workstation.

7.2.2 General Sizing Considerations

Processor: Try to follow the application vendor’s suggestions, but allow a little more than the recommended resource size. Keep in mind that the performance for a graphics application will vary from the results achieved in the vendor’s test environment.

Graphics Adapter: As a consequence of the performance scalability in a workstation, try to configure the processor/graphics adapter pair that gives you the best average response time for all the applications you need.

The RS/6000 family offers different, convenient graphics adapters to suit the needs you have in graphics display capabilities and performance. A detailed description of such adapters can be found in 4.6, “Graphics Adapters” on page 99.

Figure 77 helps you configure RS/6000 workstations.

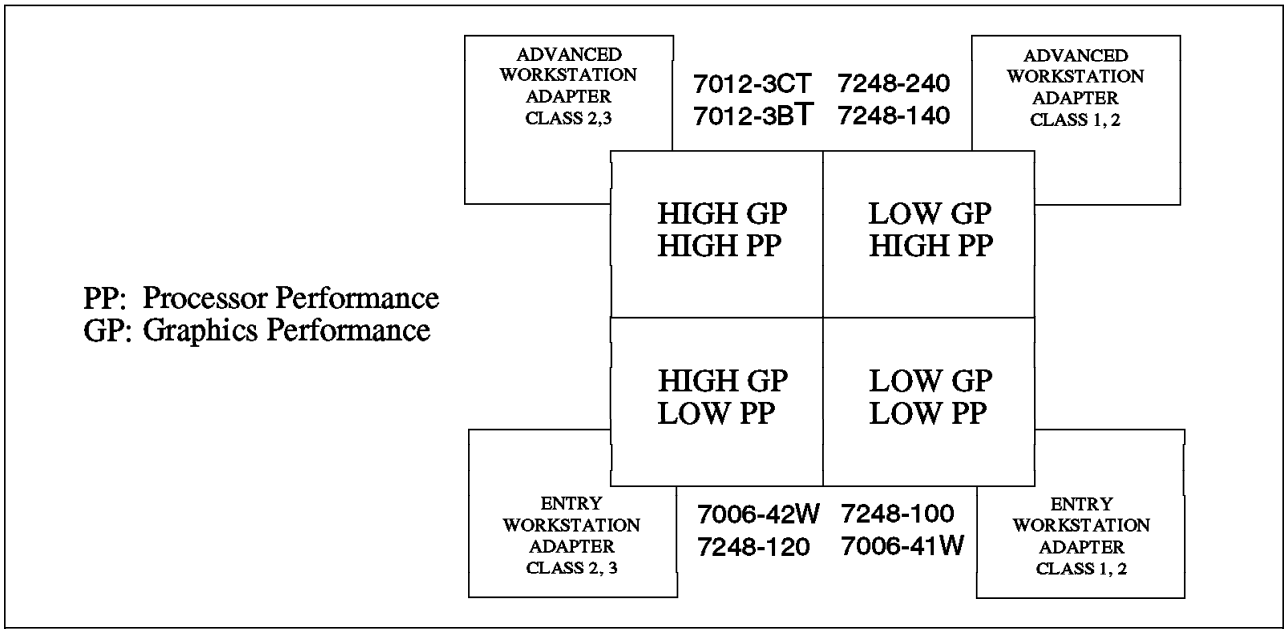


Figure 77. Processor/Graphics Adapter Selection Chart

Memory: Graphics applications demand more memory than ASCII-based applications. It is recommended that you configure at least 32 MB for a light application. Applications like GIS, SystemView or CATIA, which have high memory requirements, may require more than 32 MB.

Each new application instance in the workstation, like a user session, demands a quantity of memory following the same model as described in 7.3, “Multiuser System Sizing” on page 191. The additional quantity of memory needed for a new application instance may be equal to or lower than the initial one. Vendor’s information applies to single application instances, so if you need more than one application instance, then you need to add more memory to your system.

Disk: For entry workstations, where system and data availability are not essential, one internal 2 GB SCSI-2 drive is sufficient to store the operating system, paging space, application software and light data. Other systems may require better performance and disk capacity, so one 4.5 GB disk drive may be better.

For medium-range workstations, where high I/O throughput is needed, two disk drives offer a good solution, allowing you to distribute paging spaces and data access workload.

For workstations demanding average I/O performance and high data availability, you should configure duplicate internal disk drives so you can create mirror copies of the operating system, software and data. This gives you excellent data availability at a very low cost.

7.2.3 References

Managing AIX V4 on PCI-Based RS/6000 Workstations, SG-24-2581

http://www.rs6000.ibm.com/hardware/#workstations

http://www.eu.sun.com/solaris/products/wabi/support/wabi2.0-questions/serve2.html

7.3 Multiuser System Sizing

Multiuser configurations can serve a wide variety of purposes and experience a wide spectrum of user loads. As its name says, and by the traditional categorization, a multiuser environment is conformed by a central system executing different applications for direct-attached end users. Configurations of this kind differ highly from client/server environments in the fact that the user's program presentation, interaction and data processing are all in the central system. Only formatted data are presented on remote terminals, attached via either LAN or asynchronous serial ports.

A multiuser system can also act as a server for databases, files or other classes of client/server computing environments. This section is intended to provide some guidelines about the multiuser issue (a single system with many users). Large multiuser configurations may have users operating a variety of applications such as databases, program development, or office automation in addition to the full range of AIX commands.

As described above, the hardware to support the end-user workload depends on the application's nature, quantity of data, number of users and concurrence. The RS/6000 family of products offers a wide variety of options from entry desktop servers up to high-end deskside ones based on SMP technology. We will differentiate between uniprocessor and SMP architectures when necessary.

7.3.1 Multiuser Environment

Configuring multiuser environments depends on both application nature and user habits. In order to understand multiuser configurations, it is necessary to understand software components in a multiuser environment. Let us use a top-down approach to review software components:

- **User applications**

  There are two categories for multiuser systems, depending on how the application programs share data resources:

  - **Independent**

    AIX commands and other self-contained programs.

  - **Local clients**

    AIX programs of this type use communication methods to share data inside the same machine. All of the applications belonging to this type have a similar architecture to that of the DBMS environments. The set of

server programs running between the operating system and end-user programs is called the *application enabler level*.

- **Application enablers**

  They set the software level providing data management services to applications, such as a DBMS server.

- **Communication**

  The communication protocols and utilities are intended to allow end users to open sessions on the system.

- **System programs**

  All commands and high-level programs belonging to the operating system, such as compilers, SMIT, file utilities, and so on.

- **Support tools**

  All utilities necessary to maintain the system, support end-user tasks and keep the system performing with desirable response times.

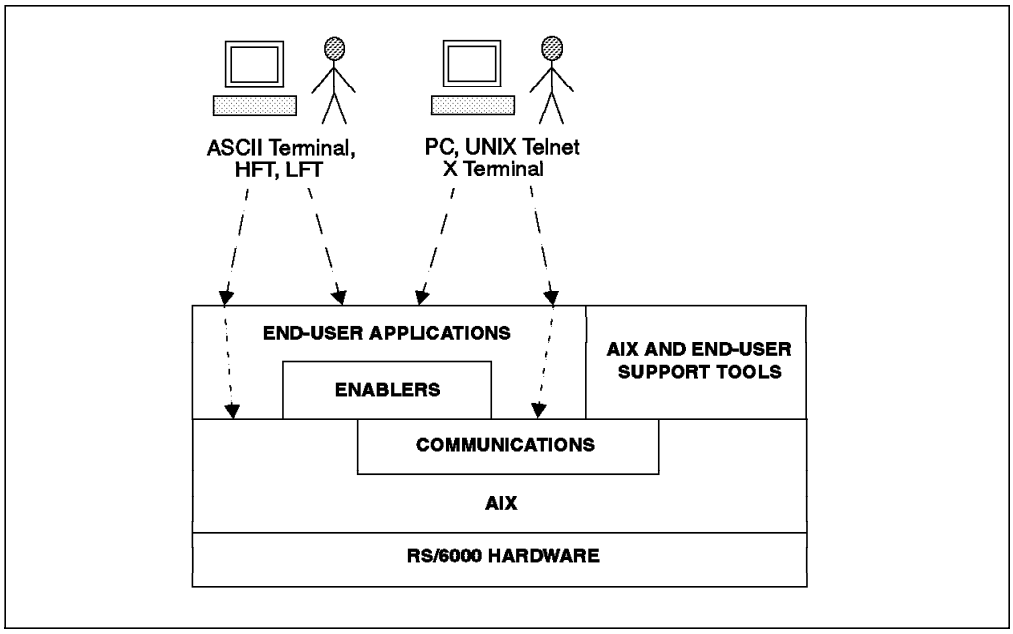


Figure 78. End-User Applications Support

It is very important to know the specific resource needs for the type of applications you want to install on your system. Independent programs demand high levels of operating system attention and system resources when the quantity of users (active and passive) is high. Independent program environments also suffer of high lock contention when many users access shared files concurrently. So, if you want to have a lot of users (70 or more) working together over the same resources, it is better to think about DBMS or a similar environment rather than independent programs.

Applications based on independent programs follow a near-linear tendency in memory consumption. This is not the same for local server-based applications. Usually, these applications require some fixed amount of memory plus some memory for each user. After a certain number of users, the server-based approach consumes less memory than applications based on independent programs.

Figure 79 on page 193 shows a comparison of memory utilization between independent programs and local server-based implementations of the same application.

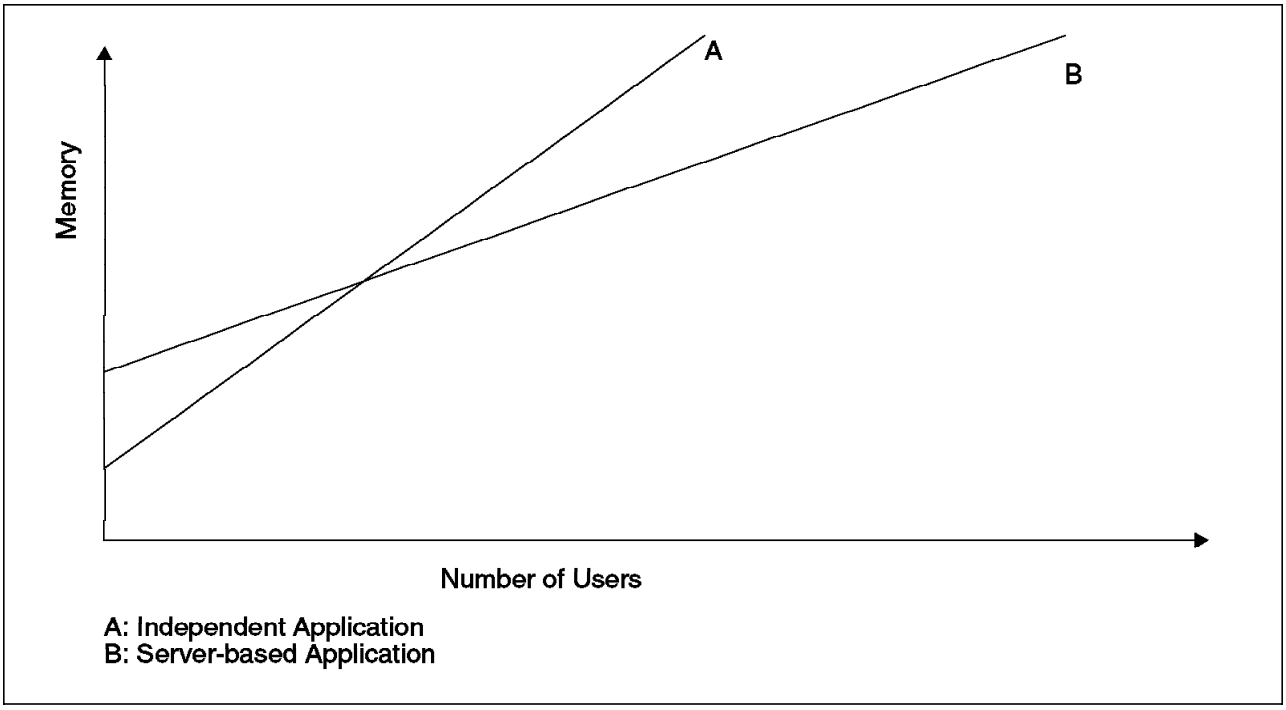


Figure 79. Memory Consumption Comparison

Using monitors or local servers in multiuser configurations helps to optimize the use of resources in the system. Many DBMS sizing considerations apply to applications based on local servers. There are architectural differences between the two implementations:

- **2N servers**

  Servers based on “2N” implementations create a server process for each local client. So, an end-user process is really representing two processes, and the optimization in system resources is minimal.

- **Multithreaded server**

  Multithreaded servers are designed to avoid the extra expense of managing many processes. Rather than having one server process for each client, there are a few specialized processes with multithreaded capabilities so that they service requests from multiple clients.

Session Support: An end-user session basically is composed of a light shell program with a prompt (where the user can start other programs or system commands) and a set of variables defining his environment. Shell demands for operating system resources are minimal. For telnet-based sessions, `telnetd` daemon resource consumption should be considered. In large systems supporting many users, the overhead caused by the asynchronous adapter devices drivers and `telnetd` daemons must be considered.

Programs that use graphical presentations through X11 libraries and the X Windows capability consume more memory and a little bit more processor power than their respective ASCII versions.

7.3.2 Workload Balancing

Sometimes, it is inevitable to have different workloads on a system. In those cases where interactive end-user, batch and computation-intensive processes fight for the system resources, it is necessary to perform a workload separation (organization). Using the AIX performance monitoring tools (as described in Chapter 8, “Performance Tools” on page 259), you can collect all the data you need to determine resource consumption for each application.

If your workload has, for instance, batch and interactive processes, then you can think about an SMP system to perform the workload separation. In some situations, it is convenient to have specialized machines for different workloads and to have good communications between them, allowing file transfer and interactive work in all of them. Consider this possibility when your response times are critical for batch or scientific applications and you do not want to see end users’ work affected.

Figure 80 shows an example of a possible workload separation. This is a particular situation where some batch processes affect the response time of interactive users in the same RS/6000 system. This figure only illustrates the portion of CPU time given to each process in the execution queue. It assumes that all applications are single-threaded. If the batch workload comprises different data and applications than the interactive workload, then it could be taken to another RS/6000 system. If workloads cannot be separated, and there is no risk of SMP lock contention, then an RS/6000 SMP system could handle both interactive and batch workloads using a batch processing queue with one or two dedicated processors.

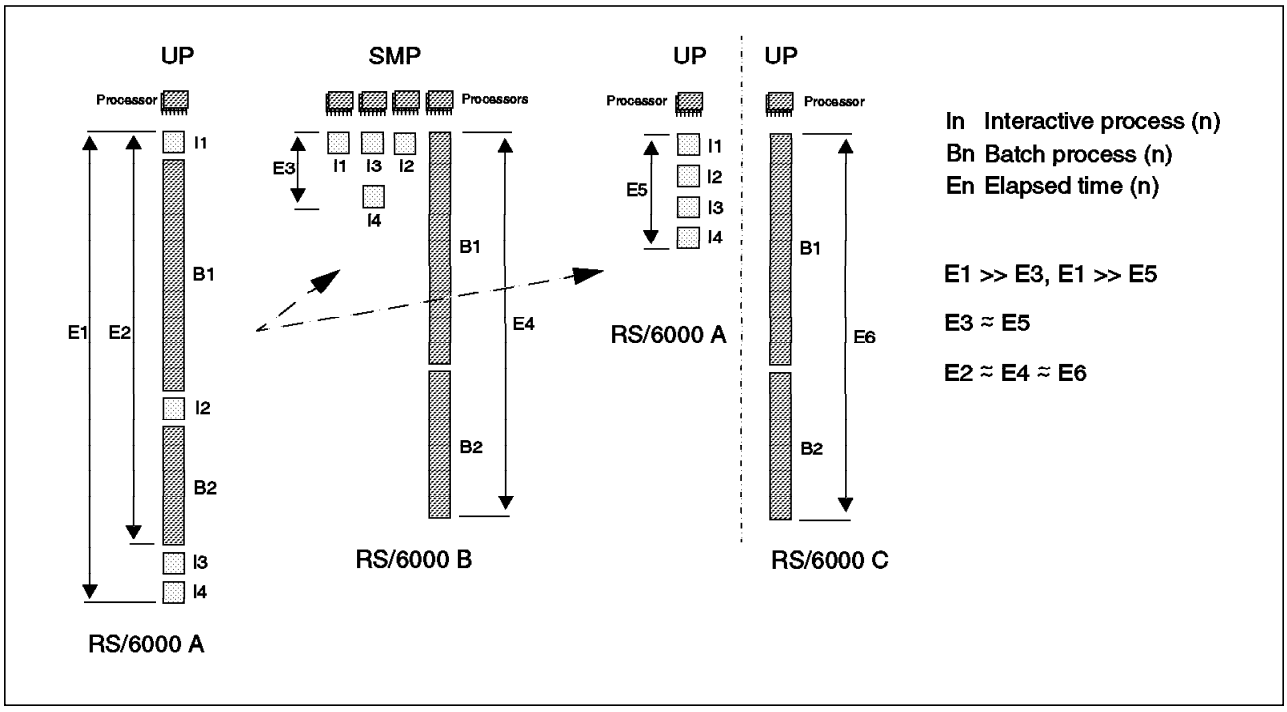


Figure 80. CPU Workload Separation Example

7.3.3 General Sizing Considerations

Given the diversity of applications, implementations, workloads, users, and requirements, it is necessary to have a detailed, in-depth analysis of an installation like the one you are sizing. Most of the time, software vendors are able to give you some guidelines based on previous installations. When sizing a server, you should also consider the resource consumption of administrative tasks such as backup and recovery, problem determination, performance analysis tasks, software maintenance, and so on.

7.3.3.1 Processor

In systems requiring very fast response times independent from the workload, you should configure your machine so that the processor will have enough idle time to handle the task (idle time > 40 percent). In systems where the average response time is the target, you should consider about 20 - 40 percent of idle time. For static systems, where no growth is expected and the workload is the same every time, the processor usage can be close to 100 percent.

7.3.3.2 Memory

The required amount of RAM in multiuser systems is dependent on the number of users, the type of applications, the type of enabler and the quantity of data the applications will handle.

Do not estimate the memory a user needs with only the size of his program code in memory. This only applies to small, self-contained programs with no external library. Most of the applications extensively use shared libraries. To be able to determine the size of memory you need, you should consult the software vendor or use AIX memory-performance-related tools (see Chapter 8, “Performance Tools” on page 259).

In order to increase performance when working with a file, AIX is able to “map” files in memory to increase access time. If there is not enough memory for buffering files, this will result in lower performance due to high I/O activity.

Do not forget that having more memory than necessary does not improve the system performance. But lack of memory will degrade performance dramatically.

Following all of these considerations, here is a convenient memory sizing formula for multiuser systems.

```
MEMORY_REQUIRED = ( Fixed_AIX_Memory +
                    Fixed_Enabler_Memory +
                    Fixed_Application_Memory +
                    Delta_Enabler_Memory( Number_of_User_Sessions ) +
                    Delta_Files_Memory( Number_of_Open_Files ) )
```

Figure 81. Memory Required for a Multiuser System

The terms in the formula are explained as follows:

Fixed_AIX_Memory: The memory needed to allocate to the AIX kernel and subsystems. This value starts at 6 to 12 MB for the operating system itself. AIX VMM also requires some memory to manage pages. This amount depends on the real memory of the system.

Fixed_Enabler_Memory: This is the amount of memory needed for the enabler code and data structures (like DBMS). This value depends on the nature and type of the enabler. Most DBMSs need from 2 to 6 MB for server binaries and from 16 to 100 MB for data structures (see 7.6, "Database Sizing" on page 211). For other kinds of application enablers, you should consult the software vendor.

Fixed_Application_Memory: Memory needed for application code and data structures. This amount is generally composed of a fixed size plus an additional size for each user. The reason is that AIX reuses the same binary code in memory for all the sessions that have loaded the same program.

Usually, this value is between 500 KB and 2 MB. Some self-contained Cobol-based and local Oracle or Informix client applications consume more than 4-6 MB. It is unusual to have applications demanding more than 10-15 MB per user.

Delta_Enabler_Memory: This is the amount of additional memory needed by the enabler to register and handle a new user. This term in the formula does not have a linear behavior. It depends on the type of enabler, how it shares resources between the user applications, and the type of applications. Enablers with "2N" design may require between 100 and 700 KB per user, multithreaded ones between 50 and 150 KB.

Delta_Files_Memory: Like the *Delta_Enabler_Memory* function, this one is used to obtain the amount of memory needed for file buffers in the system. It is very hard to find, but it is possible to use a fixed factor multiplied by the number of different, concurrent, open data files in the system. This function applies in a different way for enablers or programs using raw devices rather than JFS. Usually, 20 percent of the memory should be reserved for file buffering.

7.3.3.3 Disk

General considerations about disk storage configuration include the following:

Amount of Disk Storage For Data: In multiuser configurations, data is growing all the time. When choosing a server, you should consider the ability to increase disk storage.

Some software product specification letters talk about typical storage consumption for databases they handle. In most cases, those letters refer to basic database configurations, with just the required indexes and data structures. If you need to customize parts (or all) of an application of this kind, you might require more indexes, new tables, hash spaces for tables or data files, additional space for logical deleted records in files, reports, and so on.

In addition to data, you need storage space for operating system code, paging, program binary codes and libraries, file systems for temporary files, logs, and spooler areas.

UNIX files have an i-node-based distribution. Each file's i-node structure has a tree form, and one file is composed of many branches. When file sizes are

increasing, or updates are frequent, fragmentation becomes evident. You should configure sufficient storage space to perform defragmentation tasks.

Data Distribution: Data should be spread across disks to increase performance.

It is usually better to separate the system and paging space from user data.

For large amounts of data, disks can be organized by data types: one disk for log, one disk for indexes, and disks for data, for example.

As demonstrated by statistical analysis, typically, less than 20 percent of data causes more than 80 percent of the I/O traffic in a system. This means that you need to pay special attention to *transaction data files* and paging spaces when configuring disk drives (refer to 4.3, “Storage” on page 60).

Paging Spaces: Paging space configuration highly depends on the amount of memory and the nature of applications. The problem is not the quantity and size for the paging spaces compared with real memory; the problem is how much the paging spaces are accessed, causing disk I/O bottlenecks or demanding excessive processor time. Your applications may have good locality of reference and use the same code segment in real memory all the time. In this case, paging spaces do not need the fastest disk drives. You can follow these considerations when configuring paging spaces:

- Allow at least sufficient disk space for paging spaces to be twice the amount of real memory.

  The general recommendation is that the sum of the sizes of the paging spaces should be equal to at least twice the size of the real memory of the machine, up to a memory size of 256 MB (512 MB of paging space). For memories larger than 256 MB, we recommend:

  ```
  size = 512 MB + (memory size - 256 MB) * 1.25
  ```
- Configure only one paging space per disk drive.
- Allocate paging spaces on the fastest disks the system has.
- Use between two and six paging spaces for medium systems. Consider the possibility of having more than six paging spaces in large systems (may be in SSA disk arrays).
- Try to configure the paging logical volumes with the same size because AIX uses a round-robin algorithm.
- Do not allocate space for paging spaces in disks with high I/O activity caused by other I/O workloads.
- Configure the paging logical volumes just after installation of the operating system and always before the allocation of space for other logical volumes or file systems.

Availability: Do not forget this issue when configuring a multiuser environment. The more users in your system, the more availability it needs. Follow the indications described in 4.3, “Storage” on page 60 in order to configure:

- Disk arrays with mirroring for high performance, high availability and large storage requirements.
- Disk arrays with RAID for high-availability requirements.
- Internal disk drives with AIX and user software mirroring for system availability.

7.3.3.4 Serial Port Adapters and LAN

The RS/6000 family of products offers you different ways to connect users. If only a few users need to use your system, then you can use conventional eight- or 16-port asynchronous adapters. For more than 16 users, you should consider the 128-port asynchronous adapter.

Consider the use of terminal servers in the following environments:

- when a LAN exists.
- if you need to save Micro Channel slots.
- end users need to use more than one RS/6000 system in the network.

Terminal servers can be served by a network terminal accelerator (NTX) card when the number of telnet sessions is very high and they involve high numbers of LAN I/O operations.

For details on asynchronous and LAN adapters, see 4.5, “LAN / WAN Adapters” on page 90.

7.3.4 References

Capacity Planning for Computer Systems, Tim Browning

`http://www.govtech.net/1995/gt/jul/columns/capacity.htm`

`http://www.empress.com/services/white/white.html`

`http://www.ukshops.co.uk:8000/ap_professional/capacity.html`

`http://www.tritec.de/sunworldonline/swol-11-1995/swol-11-unix.html`

7.4 File Server Sizing

The following tips should be considered when configuring RS/6000 machines used as file servers:

- A file server should not be used as a workstation. The peak workloads of the two environments may occur simultaneously, resulting in unsatisfactory performance in both roles.
- The performance of RS/6000 file servers will generally be constrained by:
  - Processor speed, which is critical to network performance
  - RAM size for data caching
  - Disk speed

7.4.1 NFS Sizing

Dealing with the current need of sharing data, network file system (NFS) is probably the most widely used client/server application. But since it is a distributed and interactive application, and because only a few complex documents talk about this subject, configuring the right NFS server is often tricky.

7.4.1.1 NFS Theory

The NFS was developed by Sun Microsystems to allow programs on one system (the NFS client) to access files on another system (the NFS server) transparently by mounting the remote directory. This high-level function uses IP protocols for the low layers and some of Sun's specific developments, which are now de facto standards (XDR, RPC), for the higher layers.

NFS Architecture: Let us take a more precise look at the NFS architecture referring to the seven-layer network model.

- Level 7 (application): **NFS**
- Level 6 (presentation): **XDR** (external data representation).

  This is a neutral format of data. Every data exchange via NFS is made using this format, even between machines of identical architecture.
- Level 5 (session): **RPC** (remote procedure call).

  This mechanism allows execution of a procedure as if it were local to the calling process, though it executes on a distant machine. When calling an RPC, a maximum time usually is specified for the application to be able to detect an incident and to manage it (like calling back or contacting another server).

  In NFS, the incident management algorithm used by a client when a server does not respond is the following:

  – The request is retransmitted if there is no response in a certain time, specified by the `timeo` parameter (0.7 seconds by default in AIX).

  – If there is still no response after two occurrences of `timeo`, the request is retransmitted again with delay time doubling at each iteration (0.7 seconds, 1.4 seconds, and so forth, limited at 60 seconds).

  – When the number of bad calls reaches the `retrans` value (by default 3 in AIX), the message *"NFS server host not responding, still trying"* appears on the screen.

  – Then, the same procedure begins again, but with an initial value of `timeo` twice the preceding initial value (`timeo` is limited to 30 seconds). It keeps on going indefinitely until the server responds. These never-ending requests can induce a very high network traffic, and thus, influence performance.
- Level 4 (transport): **UDP** (user datagram protocol).

  This transport is stateless, unlike TCP (transmission control protocol), meaning that it gives no warranty whatsoever on the arrival of the packets. The application, in this case NFS, has to verify the information.
- Level 3 (network): **IP** (internet protocol).

  This is the standard protocol of TCP/IP.
- Levels 1 and 2 (physical layers)

  These are all the physical layers supported by TCP/IP on RS/6000, such as Ethernet, token ring, FDDI or ATM.

The first major point to remember is that NFS is based on stateless protocols.

This has two major consequences:

- **Non-transparent performance monitoring and management**

Performance of the clients cannot be measured on the server but only on the clients. The UNIX command `netstat` does not give information about which are the most resource-hungry clients, although the AIX-specific `netpmon` command does.

- **Response-time penalty**

  On average, only 50 percent of the exchanges between server and clients are real data transfers. This is due to the cache management on an NFS client.

Client/Server Dialog: Figure 82 illustrates the structure of the dialog between NFS clients and a server. When a thread in a client system attempts to read or write into a file on an NFS-mounted directory, the request is redirected from the normal I/O mechanism to one of the client's NFS block I/O daemons (`biod`). The `biod` sends the request to the appropriate server, where it is assigned to one of the server's NFS daemons (`nfsd`). While that request is being processed, neither the `biod` nor the `nfsd` involved do any other work.

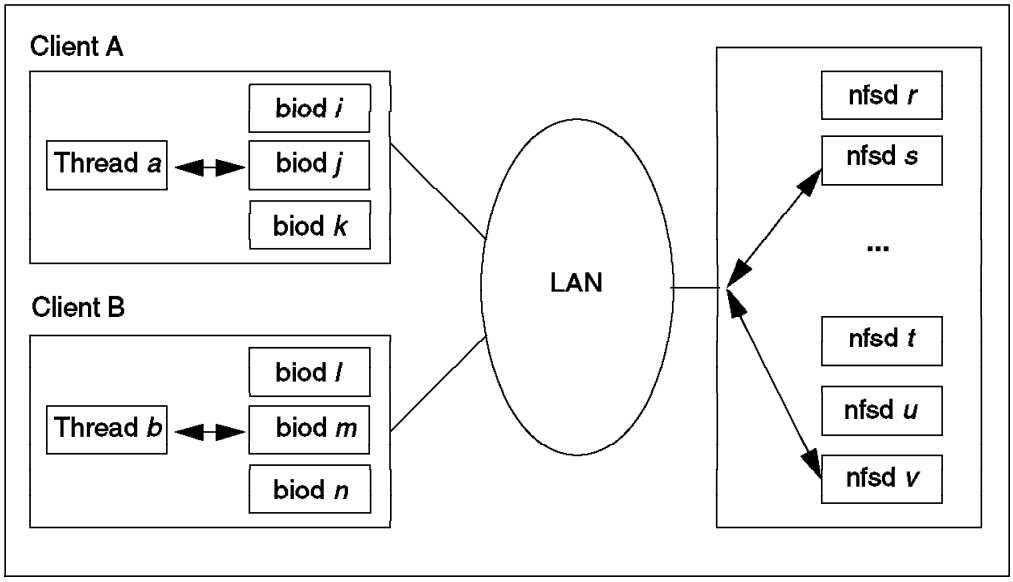


Figure 82. NFS Client/Server Interaction

Thus, in order not to have a thread blocked by lack of available `biod` or `nfsd` daemons, you must configure the right number of these daemons, knowing that:

- Increasing the number of daemons cannot compensate for lack of memory, slow processor or disk bandwidth.
- NFS daemons are cheap in memory. A `biod` costs 36 KB of memory (nine pages of which four are pinned). An `nfsd` costs 28 KB of memory (seven pages, of which two are pinned). Furthermore, an idle `nfsd` does not consume CPU time.
- All NFS requests go through an `nfsd`; meanwhile only read/write operations go through `biod`.

Cache Management on an NFS Client: In order to increase access performance to distributed files, an NFS client keeps the most-recently accessed information in its cache. The goal is to avoid other transfers over the network since the information is already on the client. However, cache coherency must be managed. As the server does not keep any record on which clients it has been

servicing, it cannot alert them when this information is modified. Therefore, it is the job of the client to manage cache coherency.

- Read access

  Each time a client accesses a file, it has to check the coherency between its cache copy and the server's original file. If the copy's last modification time stamp is newer than that of the server file, then it is considered to be good. To find this information, NFS uses the open file attributes. That data may also be in the client cache. But these attributes have a limited validity, by default, three seconds for a file (`acregmin` parameter), and 30 seconds for a directory (`acdirmin` parameter). If this time information is outdated, the client must ask for it at the server. Then, it compares it to its copy date. If the copy is older than the original, the client has to make another call to the server, asking for the data.
- Write access

  In NFS Version 2, the only way to guarantee server data integrity is to execute the operation synchronously. So, when a program needs a write operation on a distant file, the client fills the NFS buffer(s), whose size is 8 KB by default (*wsize* parameter). Once one of these NFS buffers is full, or when closing a file, the system (more precisely a background process called `biod`) generates an RPC call to execute synchronously the write operation on the server. The call ends only when the server has written on a non-volatile media.

One generally considers that an abnormally high rate of *getattr* RPC (used to find out about the file attributes) begins at 60 percent. As for response time, the biggest penalties are seen on the write accesses. Indeed, as these accesses are done synchronously, the writing and closing time of a modified file is much longer than usual.

Note that NFS data is cached in the virtual memory manager, as is any data page, but NFS data is never paged to disk space on the client. If a page is selected for pageout and later needed again, then it will require another server access to read the data.

Data Volume Transferred via RPC Call: An RPC call for file attributes transfers only a few bytes (512). A read/write operation on a file is done by blocks of 8 KB (`rsize` and `wsize` parameters) and generates one RPC call per block. Let us take a slightly worse case than the SFS benchmark to evaluate an average size of transferred packets. We can say that 50 percent of the RPC calls are read or write operations, and for 5 percent of the data transfers, RPC uses half-full buffers. (For example, while closing a file, the last buffer may be partly empty.)

$$0.5 * 512 + (1 - 0.5) * (0.05 * 4000 + 0.95 * 8000) = 4156 \text{ B/RPC call}$$

So, we will consider for the rest of this example that, on average, 4 KB are transferred on an RPC call. (SFS benchmark packet distribution would have come to the conclusion of 3 KB.) You can verify this value on your site using the `nfsstat` command.

How to Estimate SFS Results: When SFS results are not available, the most-similar available benchmark is TPC-C, since it also runs in a client/server environment though making some too-complex transactions for NFS. TPC-A would have been closer but is now obsolete.

Attention

These estimates only intend to give you an idea of the different RS/6000 servers' capabilities. They cannot, in any way, replace SFS benchmarks.

RS/6000 models	**Estimates** (rounded by 50)
C10 (1 MB L2 cache)	600
E20	900
590	900
390	950
390 (1 MB L2 cache)	1150
59H	1400
J30 (4-way)	1650
R24	1850
J40 / R40 (8-way)	7400

Table 25. RS/6000 NFSop/s Estimates

7.4.1.2 Method and Sizing Factors

1. Find an estimate of the number of NFS operations per second (NFSop/s) the server must handle by either:
   - Measuring client activity (`nfsstat -rc`).
   - Measuring server activity (`nfsstat -rs`).
   - Using some application knowledge concerning I/O throughput (remember that an NFS operation transfers on average 4 KB).
   - Applying your own experience. You may consider, for workstations, that a diskless operation generates 15 to 25 NFSop/s, a dataless 8 to 17 NFSop/s and a data-full 0 to 10 NFSop/s.
2. Find the corresponding RS/6000 server able to sustain the throughput (refer to Table 25).
3. Evaluate the network type and/or the number of network adapters, knowing that:
   - An Ethernet adapter can sustain 140 NFSop/s (about 45 percent of the maximum throughput).
   - A token ring can sustain 80 (4 Mb/s) or 325 (16 Mb/s) NFSop/s (about 65 percent of the maximum throughput).
   - An FDDI can sustain 2030 NFSop/s (about 65 percent of the maximum throughput).
   - An ATM can sustain 2650 (100 Mb/s) or 4030 (155 Mb/s) NFSop/s (about 85 percent of the maximum throughput).

4. Size the memory as follows: 32 MB is the minimum. Increase the memory size according to the percentage of files that will probably be reaccessed, in order to use locality.

$$RAM = t * T + 32 \text{ (MB)}$$

where t is the rate of pages accessed already in memory and T is the global size (or a major number) of the open files.

5. Estimate the disk space needed according to the amount of shared data and to the capability of the various types of disks. Note that an NFSop implies 0.25 to 0.5 disk operation, depending on memory efficiency when dealing with I/O operations.

7.4.1.3 Example

This example will illustrate a method of evaluating NFSop/s.

Environment: NFS clients are 40 PCs running DOS/Windows, TCP/IP and NFS. The network is a token ring (16 Mb/s). The NFS server to be determined is an RS/6000 which has the following functions:

- Application server. Software code is stored on the NFS server to save space and for easy modification.
- Server of general-interest documents.
- Data exchange among the PCs. A temporary zone shared between the clients allows them to quickly and simply exchange information.

Analysis: An analysis would show that, for each of the three functions to be performed by the server, the following data apply:

1. Five applications are used, each being 1.2 MB. The maximum load is 35 people using an application, eight of them being able to simultaneously load their software.
2. The general-interest documents database is around 100 MB and is made of documents of about 100 KB each. There is one update and 30 reads per day, five concurrent at maximum.
3. There are five data transfers in write mode per day. Each of these may induce up to 15 read accesses. At peak, four concurrent transfers can occur. The exchange volume is 500 KB.

For each of these functions, the number of NFSop/s is calculated like this:

1. If you assume that an NFS operation takes about 50 milliseconds to complete

   (SFS benchmark hypothesis), it means that a client is only able to generate 20 NFSop/s. If you want a better response time, you just have to make the corresponding calculations (for instance, if 25 ms is the goal, then you must multiply all the NFSop/s by two to size your server).

   So, for eight people,

$$NFSop/s = 20 * 8 = 160$$

The 27 remaining PCs (maximum) using the application generate much less network traffic. They only have to load the lacking pages not yet in memory. Experience shows that this kind of activity provokes less than 2 NFSop/s.

So, in conclusion, the total throughput for function 1 is:

$$\mathbf{NFSop/s\_1} = 160 + 2 * 27 = \mathbf{214}$$

2. In the same way, for five simultaneous accesses, we use the following calculation:

$$\mathbf{NFSop/s\_2} = 5 * 20 = \mathbf{100}$$

3. In the same way, for four simultaneous accesses:

$$\mathbf{NFSop/s\_3} = 4* 20 = \mathbf{80}$$

Now that we have these values, the question is: should we add all of them for peak period or not? In this case, the answer is no. The last two activities are short and done occasionally, so the probability that they will occur simultaneously with one of the others in peak times is really low. If the activities were simultaneous, the response time would decrease for a very short time only.

The second activity happens about 30 times a day and lasts about one second.

The third activity happens five times (one write and up to 15 reads) a day and lasts about six seconds.

Therefore, the system should be sized for a peak load of 214 NFSop/s.

Sizing

- **CPU**

  A small server can sustain this load, for example, a C10.

- **Network**

  One token ring adapter at 16 Mb/s should be enough.

- **Memory**

  The needed sizes for the three different types of functions are:

  1. Five applications * 1.2 MB = 6 MB. The code is accessed in read-only mode and very often by the various clients. This implies that cache efficiency should be excellent if there is no memory constraint on the server.
  2. Thirty consultations * 0.1 MB = 3 MB. The code is accessed in read-mostly mode, and the probability that the same document is accessed by the various clients is low. This implies a small cache efficiency.

3. Five transfers * 0.5 MB = 2.5 MB. The code is accessed in write mode (no cache efficiency) and in read mode (high cache efficiency, if the time between a write and the read operations is not too long). This implies an average cache efficiency.

In conclusion, a 32 MB configuration is enough for these activities.

- **Disk**

  The ratios between the number of NFS operations and the number of disk operations can be:

  1. 0.25 for the first type of activity, and probably less, as the cache efficiency should be very good.
  2. 0.5 for the second activity (bad use of cache).
  3. 0.35 for the last activity (average use of cache).

  Thus, the number of I/Os per second to sustain is:

```
I/O/s_1 = NFSop/s_1 * 0.25 = 54
I/O/s_2 = NFSop/s_2 * 0.5 = 50
I/O/s_3 = NFSop/s_3 * 0.35 = 28
```

  As before, only the first number should be taken into consideration for sizing. This means that the disk subsystem has to stand 50 I/Os per second. This can be done by every kind of disk, and only one disk drive is needed.

The chosen RS/6000 is: a C10 with 32 MB memory, one token ring adapter, and 2 GB of disk space.

7.4.2 DFS Sizing

The Open Software Foundation (OSF) originally chose the Andrew file system (AFS) as the technology for distributed file systems in the Distributed Computing Environment (DCE). AFS has since been modified to be integrated into DCE architecture and services, and AFS has evolved to become Distributed File System (DFS), available on the RS/6000 since August 1993.

7.4.2.1 DFS Environment

DFS offers many advantages over NFS.

The fact that NFS relies on stateless protocols is one of the reasons for its decreasing performance when adding more and more clients. Cache management on the NFS clients induces high network traffic. Write operations are synchronous, which hurts response time.

A DFS server keeps a record of the clients it has serviced. This allows a completely different policy of exchange between server and clients. When a client requires access to a file, the DFS server starts to send the required parts of the file. The DFS client works on a local copy, making no additional requests to the server. This is done unless data has been modified by another client. Write operations are asynchronous. Furthermore, besides a performance increase, DFS guarantees data coherency, whereas NFS may have a delay of three seconds.

The major elements in DFS are:

- **Unique image of data**

  All clients share the same data with the same root trees, names and attributes in a cell (a DCE cell is a kind of NIS domain counterpart). DCE security services make each user's authentication known on every DCE server, and a user can access all or part of the data space in a cell. This space can even be spread among several DCE cells using the Global Directory Service (GDS).

  This name coherency is not guaranteed by NFS. It is the responsibility of the system administrator, and it would impose NIS utilization. But coherency among various domains is not possible, and lack of security is one of the drawbacks of NIS.

- **Cache consistency**

  Cache consistency is maintained internally in DFS through the use of tokens. Tokens are guarantees from the file server that a client can perform certain operations on a file residing in the client cache. When the server issues the client a read data token, for instance, the client can continue to read file data in its cache without contacting the server until the server revokes that token. A server will revoke a token if another client attempts a conflicting operation.

  For instance, a second client may want to write to the area in the file from which the first client is reading. In this case, the server would revoke the read token from the first client and then grant a write token to the second client. Readers should note that if the second client wanted to write to a different area in the file, the two tokens would not be incompatible, and revoking the read token from the first client would not be necessary.

  In a nutshell, DFS implements a long-term caching capability to reduce the amount of traffic to/from the server.

  It also uses large block caching and read-ahead to speed up sequential accesses, especially for large files.

- **Availability**

  DFS supports server replicas (copy of data on different DFS servers). As for the clients, an unavailable server does not imply endless retransmissions. The DFS service will be provided by a replica of the down server, thanks to the Cell Directory Server (CDS).

  On the other hand, an NFS client cannot get any files from the server and has to wait for the server to come up.

- **Data management without interrupt**

  DFS is able to support online backups as well as data movement from one server to another without bothering the users accessing the migrated files.

- **Security**

  DFS uses the authentication mechanisms of DCE, based on Kerberos. It allows access control to resources by list (ACL: access control list). A user who identifies himself (userid, password) has a guaranteed identification of a limited lifetime and is known by all the servers in the cell.

7.4.2.2 Sizing Factors

CPU: Tests have shown that the client (and in particular, its CPU) is the limiting factor in read scenarios.

Memory: With DFS, the client cache can be configured either on memory or on disk. Both choices are useful in lessening the amount of network traffic. However, using a memory cache yields better performance in applications where very large files are used or where cached data is rarely reused. On the other hand, a large disk cache may be more appropriate in environments with heavily loaded networks, with clients that have small system memories, and/or where file data is heavily reused.

If the system is already running DFS with a disk cache, you can estimate the required size of a memory cache by issuing the following command toward the end of a peak workload period: `cm getcachesize`. Divide the number of 1 KB blocks by 0.9 to determine the memory cache size needed to accommodate the same amount of data (about 10 percent of the blocks in the cache are used for DFS record-keeping).

The size of a memory cache should not exceed 10 percent of the real memory size of the machine. The recommended size is about 5 percent of real memory.

I/O: The disk cache (if any) should not be included in the rootvg volume group and should not be on the same physical volume as the paging space. Its logical volume should be used primarily or even exclusively by the disk cache.

Network: DCE RPC uses an algorithm that allows the flow of RPC packets to adapt dynamically to the level of reliability in the network. It uses a slow-start technique to avoid overrunning the communication channel. Initially, the sender transmits data in groups of 1.024-byte RPC packets. The last packet in the group contains a bit telling the receiver to acknowledge the group. The receiver also starts with a window size (the number of packets that can be transmitted without receiving an acknowledgment) of one. As the sender gains confidence that packets are not being dropped, it gradually increases the window size and the RPC packet size.

7.4.3 Reference

AIX Versions 3.2 and 4 Performance Tuning Guide, SC23-2365

7.5 Client/Server Sizing

It is important to understand that the terms ″client″ and ″server″ refer to software, not to hardware.

7.5.1 Client/Server Environment

A software client usually consists of two pieces. The first piece is the client application software; the second is what we will call ″client enabling software″. Both communicate using a carefully specified common language, called an application program interface (API). The client enabling software takes any request the client application software makes via the API and verifies it for correctness. It then decodes the request and forwards it to one or more servers for action.

Usually, the servers reside somewhere else on a network, so the client enabling software also creates links or sessions over the network to the servers. When the servers have finished, they send the results back to the client enabling software. The client enabling software then interprets these results and gives them back to the client application software via the API. For the rest of this chapter, we will refer to the combination of the client application program and the client enabling software as the client.

A network, usually a LAN (but WAN is also feasible), carries the interactions between client enabling software and the servers. The server software can usually accept requests from dozens, hundreds, or even thousands of clients concurrently. Clients may request services from one server or from many servers, depending on the application′s needs. Client/server computing can take place within an organization or between organizational or enterprise boundaries to support a business process.

Like clients, servers are also software and can coexist on a single computer or be set up on separate computer systems. This provides flexibility.

There are five models of client/server computing:

- **The Front End Model**

  This is the simplest client/server model, where only a part of the presentation layer is distant.

  This approach provides a graphical front end for existing applications. Some call it the ″face lift″ approach, because the looks improve while everything else remains the same. The front end model increases user productivity and reduces training costs without changing the original software.

  The back end applications may provide data access services, transaction services, locking services, and other, similar functions. The applications already know how to present data to users and accept textual data from users. They format streams of data, most often block-mode screens, for output, and they interpret keystrokes as input. Other users may use the same back end applications using block-mode or character-mode terminals while workstation users concurrently use the graphical front end. Either way, the application does not know that the data is coming from another program.

- **The Remote Presentation Model**

  This second model puts the whole presentation process on the client systems.

  First, visual output generated by an application on one system gets displayed on another. Next, the system that displays the output also takes the user actions and turns them into input for the application. Several examples of the remote presentation model exist, but two approaches dominate today: the X Windows system and Web browsers.

  X11 defines a graphical interface server that runs on the user machine. The application processing is done on the client machines. The server provides a graphical output device for the client applications. It also takes the user′s mouse movements, keystrokes, and menu choices and sends them on to the correct application. Here, the client application doesn′t know how to display graphical output or grab mouse movements, so it asks for help from the server.

Another common remote presentation application is the World Wide Web. The World Wide Web is a way of interacting with data stored on machines attached to the Internet. People use software, called a browser, to look around the Internet and to retrieve data, including text, images, and video, from servers located throughout the Internet.

- **The Resource Sharing Model**

  In this case, presentation is local to the clients, and data is centralized.

  This model covers most of the client/server marketplace today. File servers, printer servers, client/server database software, fax servers, and similar products all fall into this model. The client/server software makes remote devices and data appear local to personal computer applications and users.

- **The Data Staging Model**

  Presentation is also local to the clients, but data is stored at different levels.

  Sometimes, sending all the data needed from a central site is too costly or time consuming. Replicating the data to each workstation, though, is also unwieldy in some cases. It might then be useful to duplicate the data at several sites if little of the data changes regularly. When these conditions fit, the data staging model is a good choice.

  It is fair to say this approach optimizes the costs and performance of a centralized data storage and retrieval design. It retains the elements of centralized control over the data, but it allows access to the data quickly. This makes it suitable for use, given the right conditions, for workgroups and for critical data.

- **The Distributed Logic Model**

  In this approach, neither part of the application can stand on its own. Some processing must be local and some centralized. Using the distributed logic model commits a business to using workstations, and other programmable devices, instead of ordinary terminals. This model maintains its data centrally. This makes it suitable for critical applications and data.

7.5.2 General Sizing Considerations

An added complexity in client/server environments is dealing with multiple sites.

A distributed environment needs to be studied as a group of different components, each of them needing to be examined first individually, then as a part of the global group. The different components are the network, the clients, and the server(s). You can further split these components into their hardware and software parts, like CPU, memory, I/O, operating system, and application. These various elements can be measured and will help to size the appropriate environment.

The main factors that characterize the workload of an application and which are useful to size a machine are the following:

- **Functional level**
  - Type of activity of the customer (distribution, banking, pharmacy, etc.).
  - Type of application (bookkeeping, inventory, etc.).
  - Kind of user (secretary, engineer, etc.).
- **Middleware level**

  - Type of middleware used: database, transaction processing monitor, message queuing, and so forth.

- **Application level**
  - Languages and tools used to develop the application (C, FORTRAN, 4GL, etc.).
  - Complexity of the queries in each program.
  - Complexity of the algorithmic and computational parts of each program.
  - Complexity and number of fields on each screen (determine how many characters are typed and sent between the terminals and the CPU to size the network).
  - The quality of program coding.
- **User activity**
  - Number of connected terminals.
  - Number of active users (average and peak).
  - Transactions/programs used for each kind of user.
  - Think time for each transaction/user (time delay spent by the user to think about what he is going to type).
  - Keyboard time for each transaction/user.

CPU: It is extremely difficult to offer general sizing recommendations for a client/server CPU configuration because the size is strongly dependent on the application itself.

Memory: The same can be said of memory. It is completely application-related.

Disk: The general idea is to balance the I/O workload between your disk drives.

If the application is I/O intensive, then you should consider using fast disks (like SSA), several adapters and only a few disks per adapter. You should also distribute your data so that there will not be contention on one disk while the other drives are idle.

Network: The network is a big issue in client/server environments, because your choice becomes an influencing factor on performance. It should not be underestimated, or a bottleneck will occur. You need to evaluate the mean and peak network traffic.

When using WAN, network latency may become an issue for the exchange of information between the server(s) and the clients. You have to know whether the application is interactive (which induces much traffic between client and server) or not, and you need to know what effects a long waiting time will have on the application or the users.

7.5.3 Conclusion

One of the most common client/server applications (after NFS file systems) are probably the database management systems (DBMS), which are discussed in more detail in 7.6, “Database Sizing” on page 211.

7.6 Database Sizing

There are a few widely used RDBMSs in the UNIX community, including DB2 Client/Server, Sybase, Oracle and Informix. Although they have the same goal, their implementations are quite different, which induces various sizing needs. As the aim of this book is not to provide a full-length description of all database sizing requirements, we will take a more general approach, giving common sizing guidelines.

7.6.1 Database Environment

Using a database in client/server mode means that the application and the RDBMS are not on the same system: The first is located on the front-end server, the latter on the backend server. The SQL queries are sent by the application to the database server via the network. The application may even be separated from presentation services. In this case, only the RDBMS software will be running on the user workstation. The RDBMS runs on a dedicated system to give it full access to the entire CPU capacity. Furthermore, this backend server can be optimized for the special needs of an RDBMS, which are quite different from those of client support. Reciprocally, the front-end server may also be optimized for the application and the handling of clients.

Client/server databases are not the only available RDBMS model. The stand-alone model is also used, although the server must be sized to work with both the database and the application servers, plus the hundreds of client terminals. This environment sizing will be discussed in 7.6.3.3, “Stand-Alone Database Sizing” on page 219. In terms of performance, the client/server model can really lead to performance improvements.

Parallel databases are a particular type of client/server database. There are three different hardware architectures for a parallel database:

- **Shared Nothing**

  Loosely coupled processors are linked by some high-speed interconnection. Each processor has its own memory and accesses its own disks.

  Examples of machines that implement this architecture are RS/6000 SP or clusters of systems.

  This type of architecture offers the following advantages:

  – Scalability in terms of database size and number of processors.
  – Performance gains from not sharing resources across a network.
  – Use of heterogeneous environments.

  This is the best-suited architecture for parallel queries. The query is divided among processors. The advantages are that processing is more distributed and that the database can manage a larger amount of data.

  The performance gains are almost linear if an even distribution of data is obtained. Function shipping assists in the reduction of network traffic, since functions (relational operators) are shipped to processors instead of data. This is the difference between function shipping and I/O shipping.

- **Shared Disk**

  Every processor has its own memory, but it has a global view of all the data. This can be implemented either by hardware or software.

Examples of machines that implement this architecture are clusters of RS/6000 using HACMP concurrent logical volume manager or RS/6000 SPs using Virtual Shared Disk (VSD). The natural architecture for an RS/6000 SP is shared nothing, but using a component of the system management software called VSD, it is possible to simulate a shared disk behavior. For further information about VSD, refer to *RS/6000 SP System Management: Easy, Lean and Mean, June 1995*, GG24-2563.

Possible advantages of this architecture are availability and the capacity to use a heterogeneous environment.

However, with shared disk architecture, scalability is limited. Shared disk architecture is more I/O-shipping oriented. I/O shipping is implemented by shipping data to one or more processors and then executing the database operation. It provides more movement of data because the data is transferred before any operations are performed. Another problem is data integrity. This problem arises when two or more processors try to update the same data. A global locking mechanism is needed. It can be done by either hardware or software. If hardware is used, then scalability and database size are limited. If software is used, performance may decrease since the work done by each CPU increases with the number of machines.

- **Shared Memory**

  This is also called shared-all architecture, as multiple processors access the same memory and disks.

  RS/6000 SMP servers are among the machines that implement this architecture.

  The main advantage of this architecture is performance. Since data is accessed locally by all the processors and memory is shared, the best performance results can be achieved in comparison to the other types of architectures.

  The disadvantages of this architecture are more-limited scalability and limited memory.

Transaction parallelism may be divided into two types:

- **Inter-Transaction Parallelism**

  It is achieved when different transactions are operated on simultaneously against one database. This is accomplished by having each available processor perform a different transaction.

  This type of parallelism is beneficial when there are many different concurrent transactions, none of which are heavily computational. Good results may also be obtained if the database is small but frequently accessed. Global elapsed process time is reduced.

  The hardware architecture that fits best in inter-transaction parallelism is shared memory.

- **Intra-Query Parallelism**

  A single query is split across many processors.

  The benefit of intra-query parallelism is a speed-up in processing time. This type of parallelism enables more complicated and/or more computation-intensive operations to be performed in a reasonable time span.

  This architecture is well suited for very large databases.

Intra-query parallelism can be achieved in two forms: partition parallelism or pipelined parallelism. Partition parallelism is a query decomposition. A single query is subdivided into several subqueries, each of which processes a subset of the data. On the other hand, pipelined parallelism involves dividing the query into a series of operators. The output from one operator is used as input to the other.

The hardware architecture most suited to this type of parallelism is shared nothing. Shared disk may also implement intra-query pipeline parallelism.

7.6.2 Transaction Processing Monitor Environment

A transaction processing (TP) monitor is an interface between the RDBMS server and the application. The application must be written according to the monitor's language, making transactions instead of directly accessing the database with SQL queries. Furthermore, a TP monitor must always be multithreaded.

TP monitors are generally used to provide better performance for a given configuration, or when the global workload is too heavy for an RDBMS to manage it on its own. They supply better performance because they regroup many client requests into one directed to the database server. Thus, fewer processes or threads are used on the server side, and there are fewer concurrent RDBMS clients, which means less demand on resources, particularly memory. Moreover, it also implies less network traffic, which is essential when clients and server are connected through a slow WAN or a busy LAN.

7.6.3 Database Server Sizing Method

Foreword

The guidelines that will be developed in this section are very useful for optimizing your server configuration.

Nevertheless, application and RDBMS efficiencies are much more important to the performance of the global system. This means that no matter how well your server configuration is optimized, if the application queries or indexes are not carefully designed, you will lose much of the performance improvements.

We are going to review the major points to determine during the sizing process.

- **Type of RDBMS Architecture**

  There are two kinds of database architectures. The first one uses two processes per client connection, one on the client system and the other on the database server. When the number of clients increases, the number of processes on the server system also increases, which adds to the load on the CPU. This architecture is used by Informix On-Line Dynamic Server Version 5, DB2/6000 Version 1 and 2, and Oracle Version 6.

  The other architecture is multithreaded and, thus, needs only a few processes on the server that will dispatch a thread for each client connection. This architecture is used by Informix On-Line Dynamic Server Version 6 and higher, Sybase SQL Server 10 and higher, and Oracle Version 7. It is planned for DB2 Client/Server.

- **Type of Application**

  You can split the applications into two kinds:

  – Classical transactional application

    Transactions are simple; there may be a few batch jobs. No heavy transactions are involved. These transactions should execute on an RS/6000 in a few hundredths or tenths of a second.

    Such transactions form an on-line transaction processing (OLTP) workload, a continuous series of short-running transactions. Batch jobs are long-running loads that should not be underestimated. If possible, batch jobs should be started during the off hours so as not to disturb online transactions. Otherwise, you should add their requirements to those of the OLTP workload.

  – Application with resource-intensive queries

    These include data mining or decision support. A single transaction can take as long as a few hours to execute on a system. This type of transaction is very complex, with many multiway joins. It generally involves huge amounts of data (from 10 GB to several TB).

- **Number of Users**

  It is very important to differentiate the total number of users that may be connected to the database from the number of concurrent users actually working on the database at the same time. By concurrent users, we mean people typing in data as well as those connected users who may just be thinking at their keyboards for a few seconds.

  The number of active users a system can support can be estimated from the throughput, average response time, average keying time and average operator think time by LITTLE′s Law:

$$\text{Number of Users} = \text{Throughput} * (\text{Think\_Time} + \text{Keying\_Time} + \text{Response\_Time})$$

  Assume that we have the following average times:

  – Think_Time = 15 seconds
  – Keying_Time = 10 seconds
  – Response_Time = 5 seconds

and that we found that the system has the capacity to handle 11.000 of these transactions per hour (or three per second).

1. We can deduce that the system will be able to support a number of users equal to: 3 * (15 + 10 + 5) = 90 users.
2. Suppose that we would like to know how many users this system can support with users who are keying more slowly (for example, a keying time of 20 seconds): 3 * (15 + 20 + 5) = 120 users.

Thus, for a given configuration, we can see that the number of users that can be supported is totally dependent on the characteristics of both user and application.

- **Development Tools Utilized**

  Find out whether a classical development language (C, Cobol, etc.) using SQL or a 4GL (sqlforms, 4glwindows, and so on) is used.

- **Database Size**

  Ask the customer for the raw database size (at least an estimate is necessary).

- **Workload Estimate**

  It is essential to try to establish an estimate of the load the server will have to stand. Information like the peak number of transactions per hour and the number of tables to join for a single query help with this approximation.

7.6.3.1 Classical Transactional Applications

Attention

All numbers are estimates found with experience, but if you ask for your database vendor's own numbers, they will be more accurate.

CPU: Always remember that you have to size the server for the peak workload that can occur, not for the mean one.

An IBM internal benchmark called relative OLTP performance (refer to 6.4, "OLTP" on page 170) is given for all servers in the RS/6000 family. It helps compare the different RS/6000 servers among themselves. You might also use the TPC-C results with extreme care, because TPC-C probably did not run the same workload as your application.

Table 26 on page 216 shows the number of users a server can handle with DB2 V2.1. The workload used is no official benchmark and represents a light OLTP environment. These numbers are just quoted to give you an order of magnitude. They are no guarantee whatsoever of a certain level of performance with your particular application.

RS/6000 Models	25S	C10	C20	E20	390	39H	59H	R24
OLTP Users	88	142	181	215	263	292	388	429
Memory (MB)	120	192	248	296	360	400	528	584

Table 26. Users for OLTP Server

Memory: Memory is one of the most crucial factors influencing RDBMS performance. Indeed, as explained in 2.2.3, "Memory Management" on page 8 and 4.2, "Memory" on page 48, access to memory is many times faster than to disk, and since RDBMS accesses data for all its operations, the less disk I/O, the better.

Thus, the addition of any level of cache is likely to improve the performance if the application reuses some data (refer to 2.2.3, "Memory Management" on page 8 and 4.2, "Memory" on page 48).

If you cannot obtain enough detailed information on the application and its corresponding RDBMS, you should overestimate memory.

Memory can be divided into several parts.

$$M = M\_Aix + M\_Rdbms + (Nb\_Users * M\_User) + M\_Cache$$

where

- M_Aix is the memory required for AIX (at least 16 MB, 32 MB is better)
- M_Rdbms is the memory required for RDBMS executable and application binaries (4 to 20 MB).
- Nb_Users is the number of concurrent users.
- M_User is the memory required for the RDBMS code per user (500 KB to 1 MB per user for non-multithreaded database architecture, and 100 KB for multithreaded architecture).
- M_Cache is the RDBMS data cache. If possible, allow at least 1 percent of the total data size, up to about 10 percent. Experience shows that, generally, about 90 percent of the data retrievals deal with only 10 percent of the data.

It is advisable not to configure less than 64 MB of memory for database servers.

Disk: First, you need to provide a sufficient storage capacity to your system, then you must try to optimize it.

As for the storage, knowing the raw database size, you must add to it logs, indexes, RDBMS control data, temporary space (for sorting, etc.) and maybe mirrored data for better availability.

This can lead to twice the raw database size, depending on:

- How many indexes are created.
- Which log/archive policy is chosen (how much history you want to keep).

- The amount of control data the RDBMS uses (database scheme, and so forth).
- The temporary space needed for sorting or operations on tables.

When implementing mirroring, you can even reach three times the raw database size. As a general recommendation, we would suggest about twice the raw database size if you lack any precise information. Extra space might always be used to better distribute data and therefore gain some performance.

Concerning performance, here are a few guidelines:

- Unless you are working with a very large database, there is no need to choose the biggest disk drives. You are better off choosing a setup that will allow you to distribute your data across as many drives as possible (cost being the only limit).
- As a general recommendation, you should not put on the same hard disk any of the following:
  - DB data
  - DB indexes
  - DB logs
  - Mirrored DB data
  - DB sort and work spaces
  - Application binaries
  - AIX operating system
  - Paging space

  This implies four drives for AIX, data, indexes and logs as a minimum. Indeed, at each update, data will obviously be changed, as well as indexes and logs. Therefore, having each of these on different drives will allow these operations to take place in parallel. If you are looking for performance improvements and some tables are very update-intensive, then you ought to split these tables across many drives, no matter how empty these drives could be.
- Concerning SCSI buses, initially choose SCSI-2 fast and wide. But even if 15 devices are allowed on this kind of bus (or 7 on the narrow SCSI), do not attach more than four drives per SCSI adapter/bus. Otherwise, you may saturate the bus (only 20 MB/s of throughput) either with data or flow control.
- The best choice for performance should be the SSA disks (from the serial architecture).
- You can use disk striping for further-improved performance, but this is not compatible with AIX mirroring.

Last, but not least, most RDBMSs can work with either file systems or raw disks (data directly managed by the RDBMS). For performance reasons, it is better to choose raw disks, as less system overhead is induced, although file systems are easier to use and more flexible. Furthermore, file systems can be mirrored using AIX, which is not the case for raw devices.

Another important factor in I/O performance is disk striping, which is available with AIX Version 4 (refer to 4.3.9, “AIX Logical Volume Manager Performance and Sizing Concepts” on page 78). As explained there, it will result in higher performance, especially for serial accesses (which is the access mode used by logs or archives, for instance).

Network: A few general guidelines:

- In case of a great number of clients, you should separate the network used between clients and the application server from the one utilized between the application server and the RDBMS system.
- If there is a great amount of exchanged data (several hundred megabytes), you should consider ATM (both LAN and WAN) or FDDI (LAN) instead of any other, less-efficient network.
- LAN

  Because of its collision mechanism, Ethernet may not be able to sustain more than about 50 clients. On the other hand, a token ring should be able to sustain more, like 100 clients. You should not try to configure too many users on a LAN because you might saturate it. (Refer to 4.5, “LAN / WAN Adapters” on page 90 for a selection of available hardware LAN adapters.)
- WAN

  If you have to put RDBMS and application servers in different locations, then you should consider implementing between them either a high-bandwidth leased line or an ATM link.

  Another recommendation may also be to use a TP monitor, as it helps reduce network traffic (among its other advantages).

7.6.3.2 Heavy-Query Applications

These applications are very hard to size in general terms. Each query has to be analyzed to evaluate its requirements. For instance, the creation of the right index may yield a dramatic improvement in the time spent by a certain query (from several hours to a few minutes).

CPU: You might use the TPC-D results with extreme care, as TPC-D probably did not run the same workload as your application.

Depending on your application type, you should choose the best corresponding hardware according to the explanations given in 7.6.1, “Database Environment” on page 211 on parallel databases.

Oracle Parallel Server has chosen I/O shipping, whereas the other parallel databases (like Sybase or DB2 Parallel Edition) have chosen the function-shipping mechanism.

Memory: The same principles apply as those explained in 7.6.3.1, “Classical Transactional Applications” on page 215.

Disk: The sizing guidelines are similar to those explained in 7.6.3.1, “Classical Transactional Applications” on page 215. You also should consider specific requirements of a parallel database. For example, DB2 Parallel Edition recommends that no more than 10 to 15 GB of data be attached to one node.

Network: Since the different machines in the cluster need to exchange much information because of the parallel database architecture, it is recommended that a private network be provided linking only the systems running the parallel RDBMS. An Ethernet or a token ring is likely to become a bottleneck to this kind of cluster. You should choose ATM or FDDI.

A network is also required between the clients and at least one server within the cluster. The speed of this network may not be as critical as that of the private one, unless fetches containing a large number of lines are frequently required.

7.6.3.3 Stand-Alone Database Sizing

In the stand-alone architecture, clients are connected to an RS/6000 running both the RDBMS and the application servers.

CPU: The CPU used in a stand-alone system has to be more powerful than CPUs used in a client/server configuration, since it has to handle all kinds of instructions.

Memory: More memory has to be configured because there will be many more processes running on the server.

Referring to the client/server section, memory may be sized as follows:

$$M = M\_Client/Server + Nb\_Users * M\_Prog$$

where

- M_Client/Server is the memory required for the same environment but in client/server (refer to 7.6.3.1, “Classical Transactional Applications” on page 215)
- Nb_Users is the number of concurrent users
- M_Prog is the memory required for each user program (0.5 to 0.7 MB per user if C programs, 1 to 2 MB per user if 4GL programs).

Disk: The sizing is similar to the steps explained in 7.6.3.1, “Classical Transactional Applications” on page 215.

Network: The sizing is similar to the steps explained in 7.6.3.1, “Classical Transactional Applications” on page 215.

7.6.4 Conclusion

You must be cautious when using the numbers given throughout this part, as they will depend on:

- The type of database you will be using: serial or parallel.
- Which database you will really have to implement.
- Whether you will use a TP monitor.

Moreover, these numbers are relevant today with the current versions of the RDBMS but will probably soon have to be reviewed.

Last, but not least, if using TPC results to extrapolate the required CPU, be aware that this benchmark is written in 3GL instead of 4GL (commonly used in real situations with customers) and that there are additional servers between the measured one and all the clients. These factors help reduce the load on the tested server but do not reflect general situations.

7.6.5 References

DB2 Parallel Edition for AIX/6000 Concepts and Facilities, SG24-2514

Relational Databases: An AIX Client/Server Approach to OLTP, by C. Gallisa, AIXtra, July/August 1993

Sybase for IBM RS/6000 SMP servers, by J. Bersin (from Sybase Inc.), AIXtra, February 1995

Informix On-Line Dynamic Server, by D. Clay (from Informix Software Inc.), AIXtra, February 1995

DB2 for AIX, by G. Kligerman and M. Wu, AIXtra, February 1995

Benefits of Transaction Processing over RDBMS, by D. H. Brown Associates Inc., AIXtra, 1994

Performance monitoring and capacity planning, by H. Kern and R. Johnson, on the Web:

http://www.sun.com/sunworldonline/swol-11-1995/swol-11-unix.html

DB2 Family: Client/Server Performance Measurement Series, by R. Gilles, T. Munk and H. Smith, on the Web:

http://stls1dv7.stl.ibm.com:80/users/m60/www/db2_cs_perf.htm

IBM Software Servers for AIX Capacity Planning Guide, by D. Raddatz and G. Sullivan, IBM Intranet:
http://w3.austin.ibm.com//afs/austin/depts/e54s/public_html/reports/eagle/eperfrpt.html

Exploring IBM Client/Server Computing, by David Bolthouse, edited by Jim Hoskins

7.7 Xstation/Diskless/Dataless Sizing

In the Xstation/diskless/dataless environment one or more hosts download necessary data to the connected Xstation/diskless/dataless workstations to perform the startup and client/server process.

To perform sizing in this environment, we have to take care about two sites: the hosts site and the Xstation/diskless/dataless site.

7.7.1 Xstations

Xstation is a particular name for the IBM X terminals. Xstations are generally less expensive, easy to use, quiet and easy to maintain. An Xstation is essentially a remote graphics terminal. Typically, it has two specialized processors, one for graphics and the second for network support (it may have a local disk for fonts), and it relies upon the server for all conventional CPU, disk, and memory resources. The Xstation connects to the host system through the LAN connection (Ethernet, token ring).

Xstations provide many of the functions of locally attached graphics terminals via remote, LAN-connected displays. The Xstation is based on X Windows, which is a model of distributed processing, namely the client/server.

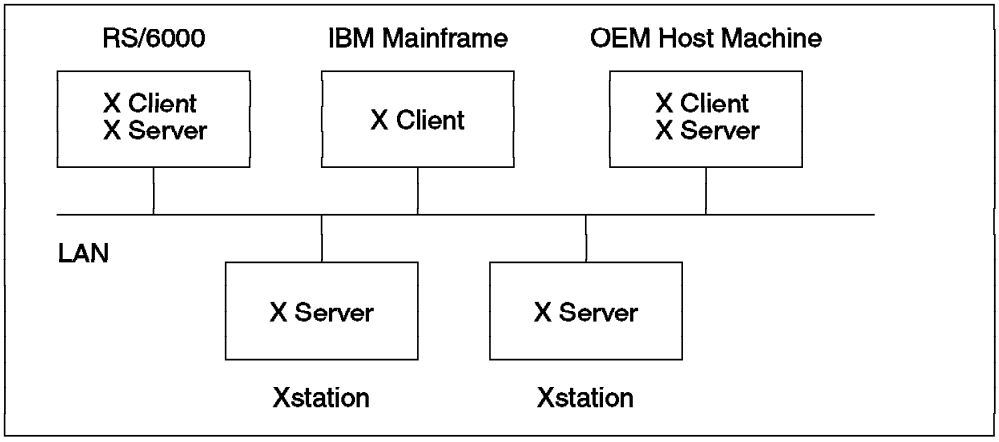


Figure 83. X Window System

The X terminal (Xstation) is a machine that executes the X Server part of the X Window system. The X Client will run on the CPU of the host system.

The X Server has several functions:

- Manage the output to the monitor
- Manage the keyboard and mouse input
- Manage the queues
- Handle the communication between the X Server and X Client
- Store screen data
- Download fonts

It is important to differentiate this terminology from traditional client/server applications such as databases or file servers, where the host system that contains the data to be shared is considered the server and the systems that share the data are considered the clients.

A single Xstation attached to an RS/6000 workstation can offload the graphical output processing on the host system because the application workload is run on the host system and the graphical output is run on the Xstation. The only limiting factor would be the LAN overhead. As the number of Xstations increases and the host system is required to support multiple X Servers, resource utilization on the host system can increase dramatically.

Limitations: Xstation performance is dependent on the following factors:

- **Application**

  Applications that do complex graphics are limited by the ability of the Xstation to perform the graphical renderings. Benchmarks such as X11perf and Xmark can provide a good basis for determining the raw capabilities of the various Xstation models. The Xstation is intended to run 2D applications; 3D applications need a high-bandwidth data transfer between CPU, memory and graphical subsystem that is much higher than the LAN bandwidth.

- **LAN**

  Xstation configurations that send large amounts of data such as bitmaps, color maps or graphical models can saturate the LAN. A good example of this type of application would be computer aided design (CAD). The network is also a performance consideration during boot time of the Xstations.

- **CPU**

  One X Window typically generates the same load as a telnet or rlogin session and will consume twice the CPU resources of a locally attached async session. All input, whether a single character or text, generates host CPU load. This ranges from the extreme case where an application does polling of the cursor position to a simple case of a keystroke. In addition to this overhead, the application itself requires CPU resources.

- **Xstation RAM**

  The Xstation has three types of configurable memory:

  1. Flash memory, which holds the X Server, utilities, diagnostics, and fonts. Some Xstations use it to store the X Server code, rather than downloading it over the network. This results in faster booting.
  2. System memory, which holds the X structures, colormaps and fonts. The enhancements of adding system memory come into play when the application program takes advantage of the backing store, save under, and off-screen pixmap features.
  3. Video memory, which holds the pixmaps or images. Increasing the size of the video memory will allow for better-quality graphics (resolution and more colors), but because the image has to be transferred from a host to the Xstation, network overhead is increased.

  The size and number of bit planes and colors is determined by the size of the video memory for the Xstation and can only be modified by adding or deleting memory. Configuring an Xstation with additional RAM will enable the Xstation to keep complete X structures and colormaps locally, thus reducing the LAN and memory requirements on the host.

- **Host System RAM**

  Enough memory must be configured to support both the application and additional X overhead. X needs to maintain the context of each window session, and thus the memory requirement can be substantial.

Xstation Sizing: An Xstation imposes a much greater CPU load on the RS/6000 workstation to which it is attached than an ASCII terminal does. In most Xstation configurations, the host can be fitted with enough RAM and disks so that the CPU becomes the limited factor on the number of Xstations that can be supported. In general, the number of Xstations that can be supported by an RS/6000 host ranges from 1 to 40, depending largely on the requirements of the

application, the speed of the processor and the amount of RAM. If the application is 3D solid or surface modeling, only about one user can be supported by an RS/6000 processor, regardless of the speed of the CPU.

X Windows is a heavy consumer of both CPU and main memory. Because of this, the number of users that can be supported on Xstations is not comparable to that of ASCII terminals. To give a broad overview of the CPU utilization caused by an Xstation user, consider this:

- The total CPU utilization on an RS/6000, caused by a single process running in one open window, in a few arbitrarily chosen test cases, was found to fall between a low of 4 percent for a character-based application to a high of 15 percent for a desktop-publishing application. Further investigation revealed that 50 to 80 percent of the time was spent in system mode.
- Because of the significant percentage of time spent in the kernel, in particular, inside device drivers, interrupts are disabled for long periods of time -- approximately 3 to 6 percent of the time for each active X Window. Because of this fact, supporting numerous concurrently active Xstations from one machine might lead users to perceive system response time as unsatisfactory, even when the CPU is not yet saturated.

There is a formula available that gives a more accurate estimate for the CPU utilization of a particular application than the figures quoted above. This assumes detailed information about the application is available.

Let:

A = CPU seconds absorbed by the application per minute.
K = Number of keystrokes per minute.
C = Number of ASCII characters written to Xstation per minute.
G = Number of X11 calls per minute.
V = Number of vectors drawn per minute.

Then, Xstation consumption in milliseconds of CPU time on an RS/6000 per minute is:

```
CPU Time = 1000 x A + 14 x K + 0.09 x C + 0.5 x G + 0.25 x V
```

Figure 84. Xstation CPU Consumption

The constant factors in the formula above scale to other CPU speeds, except A. But even for very fast CPU speeds, this formula should give you a good estimate.

The second point of concern in sizing a configuration for Xstations or a natively attached low function terminal (LFT) is the memory requirement of X Windows. Per Xstation, X Windows consumes a minimum of 4 MB, mainly to hold the frame buffer which stores the pixel image of the screen for the X Client. The typical memory requirement lies in the 4 MB to 8 MB range per user. This figure is heavily dependent upon the number of open windows. Expect 360 KB to 550 KB private virtual memory usage per open window in addition to the base requirement of 4 MB. These pages are likely to be paged out for inactive windows. The Motif window manager program needs, typically, an additional 500 KB of private memory per user.

It is recommended that a configuration consists of many low-end hosts rather than a few high-end hosts. For instance, given a choice between one RS/6000 in a client/server environment with 20 Xstations, or three smaller RS/6000s as hosts with a total of 20 attached Xstations, the latter configuration is preferable.

Because the processor will tend to be heavily loaded, a system with Xstations connected should not also be used as a workstation or file server.

On the host site, sizing depends on:

CPU The type of application, number of concurrent users, and extent to which the application uses the windowing facilities should be considered when selecting the host CPU.

RAM Allow 4MB to 8MB of additional RAM in addition to the application and system requirements, per Xstation. The workload will dictate the amount of RAM that is needed. Applications that use lots of fonts and shuffle windows back and forth need more RAM than those that do not.

DISK Systems that require significant amounts of application data should be configured in such a way that the data is optimally distributed over the disk drives. Estimating the disk utilization of the application(s) can be done by comparing the projected I/O rate with the capabilities of the various disk drives.

LAN The most important rule on networking is “use the best adapter possible”. The network load depends on the application(s) that the Xstation is running. If the network traffic is the problem, you can divide the Xstations into multiple networks by installing more LAN adapters in the hosts.

Other factors Xstation sizing depends on:

- For the Xstation itself, there are no options on CPU, since the Xstation has auxiliary graphics and communication CPUs that handle display and networking protocols only.
- On the Xstation itself, the 30 MB hard disk is for storing files that would otherwise have to be transferred from the host at boot time. If the application uses complex graphics and lots of images and fonts, a disk to reduce the load on the network and local RAM may be worthwhile.

7.7.2 Diskless Workstations

A diskless workstation is just like a regular RS/6000 system but without any local disk drive. This means there is no file system, no boot image, and no paging space. The diskless workstation obtains resources from other machines in the network. The install images or LPPs for the diskless system are shared with the server that contains the operating system of the diskless workstation (*/usr/share*). The server is an AIX or UNIX system that can provide the boot image, file tree, and paging space for the clients (diskless workstation).

Despite the fundamental differences between Xstations and diskless workstations, they are often considered as equivalent, interchangeable components in a client/server environment, particularly, when cost is a big factor. Although similar in appearance, they differ in the kind of applications they support well. It is important to understand the underlying principles and technologies of these two products in order to recommend them wisely.

The characteristic that makes a diskless workstation unique is that although it has all the functions of a stand-alone workstation, it must rely upon a server for all disk I/O requirements, including paging I/O. Local memory and CPU can give the diskless workstation a significant amount of independence from the server. Given enough memory to support the working set of an application, the diskless workstation can provide the performance of a stand-alone workstation at a smaller cost.

Unlike Xstations, the amount of memory in the server is not a major factor. Tests have shown that as little as 64 MB of server memory may be sufficient to support a large number of diskless workstations.

NFS in a Diskless Environment: In the diskless environment, each client depends upon a server to provide disk I/O. This is made possible through the network file system, or NFS, which makes remote files appear to be local. NFS is stateless, meaning that each client′s request is independent from any previous or subsequent communication. Once fulfilled, each operation is promptly forgotten by the server and client. This keeps the protocol simple. Deadlocks in which a server or client is kept waiting for a response that never comes are avoided.

However, there is a drawback to this scheme. Each write to an NFS file system must be completed before the client initiating the I/O request may proceed. Writes of this type are called synchronous. This delay is needed to ensure that data to be written to an NFS file system is physically committed to disk and not just buffered in the server′s memory. This is often the case with local file system activity. In AIX, each NFS write operation involves multiple physical writes: data, the file system journal, and the metadata. Clearly, writing to an NFS file system is a time-consuming operation. This I/O constraint cannot be overlooked when a diskless workstation solution is being proposed.

In particular, be aware that the diskless workstation uses NFS as a paging mechanism. A simple rule of thumb is that applications that exhibit high paging rates are not suited for the diskless environment unless enough memory is available in the diskless workstation itself to keep the paging accesses to a minimum.

Limitations: Diskless workstations are an inexpensive way to increase computing power. There are some important guidelines that must be followed in order to configure them properly.

- Client RAM

  Each client must have enough RAM to contain the working set, where working set is defined as the text, data and stack segments of an application. Working storage paging is to be avoided at all cost. The penalty of paging across a network is so severe that it will often make response times or throughput unacceptable.

- Disk Layout on the Server

  Since disk I/O is usually the limiting factor in a diskless environment, careful thought should be given to the server′s disk configuration. Obviously, if the application being studied does frequent I/O involving large amounts of data, there must be ample space on the server. Several smaller disks are preferable than one large one, so that each client′s heavily used file systems and paging space can be placed on different physical volumes. The logfile should also be placed on a separate disk when possible.

Each client needs its own private space on the server. Do not make this space too small, since doing so is a false economy.

Aggregate disk utilization should not exceed the 40-percent mark. If the utilization exceeds this boundary, consider reorganizing the file system layout or adding more physical volumes.

- Network Topology

  With minimum planning the network will give efficient service to clients and servers alike. Ethernet utilization should be in the range of 35 to 50 percent. The latter value is valid when there are 15 or fewer clients on a network. High-performance adapters should be used. The physical length of the cable should be kept at a minimum to avoid detrimental electrical characteristics and excessive collision rates. If an IBM token ring adapter is employed, the utilization figure may safely rise to 70 percent.

- Server CPU

  Because remote file systems impose such a heavy burden on the server's CPU resources, it is important that it is fast enough to provide an acceptable response time. There are many layers of software involved in serving clients with disk I/O. Some of the more important ones are the internet protocol (IP), user datagram protocol (UDP), NFS, and the appropriate device driver and file system code. Read and write requests are very costly in terms of CPU resources. The server CPU must be powerful enough to do the job.

Diskless Sizing: The key to a successful diskless installation is configuring each client with enough RAM to contain the working set of the application so that an I/O bottleneck is avoided. Some initial page fault activity when the program and data tables are loaded cannot be avoided, but once they are resident in memory on the client, there should be very little paging.

The best way to plan a client/server installation is to measure the workload. A realistic simulation is one way to do this; an even better solution is to benchmark the proposed application. Whichever method is chosen, it is important that the peak workload, not the average, is considered. For example, a server may be busy only 15 percent of the time when averaged over an eight-hour day. At peak times, usage may be much higher. If the configuration is tailored for the average workload, performance during the peak times may be unacceptable.

On the server side, sizing must focus on:

CPU The server's CPU must be capable of performing the required network and disk access fast enough to satisfy the throughput requirements. This overhead can be avoided to a large extent by configuring the client system with the proper amount of memory. Given enough memory in the client system, the speed at which an application runs will be dictated by the client CPU and not the server.

RAM Servers managing large files or a large number of files may see greater improvement with more physical memory. The server cannot cache writes but can cache reads. More memory means fewer disk reads, which improves response time.

DISK Systems that have a requirement for significant amounts of application data should be configured in such a way that the data is optimally distributed over the disk drives. Estimating the disk

utilization of the application(s) can be done by comparing the projected I/O rate with the capabilities of the various disk drives.

In addition to these requirements, the server disk subsystem should be capable of accommodating the paging space and the boot image for each client.

LAN The network considerations are the same as for Xstations. Additionally, you have to consider NFS paging and the possibility of a change of the diskless station into a dataless station. Applications that do sequential I/O on large files should be run on the host not on a diskless workstation. Each diskless client needs an NFS daemon.

On the client site, the sizing must be focusing on:

CPU Given enough memory in the client system, the speed at which an application runs will be dictated by the client CPU and not the server.

RAM The entire working set of the client's application must be contained in memory to achieve good performance. Therefore, the proposed workloads should be studied to determine how large the working set is.

The size of the memory is the most important parameter for the diskless site. Working storage paging over the network is very slow, and the performance will not be acceptable. Most applications have recommendations against it. Also consider, by running multiple windows under X at the same time the requirement for more memory will also increase.

7.7.3 Dataless Workstations

The environment for the dataless workstation is similar to a diskless environment. It differs in the type of dump and local paging space it uses. Diskless clients use NFS to access remote dump and paging devices. Dataless clients use dump and paging devices located on a local disk.

Performance can be improved and network traffic decreases if the diskless client is converted into a dataless client. This can be done connecting a disk to a diskless client and configuring the disk for use.

7.7.4 Case Studies

I/O-Intensive Workload: It is reasonable to consider either an Xstation or diskless workstation for situations where low-cost workplaces are important. Careful analysis of the proposed application, or workload, is required to make the right choice. The critical factor in making this decision usually is the amount of network traffic the workload will generate.

A diskless workstation uses the network file system (NFS) to make remote file systems appear local. Workloads that are I/O-intensive will place heavy demands upon the network. The network subsystems will frequently be called upon to shuffle data and programs back and forth between the diskless workstation and the server. The system demands may change drastically when this kind of workload is running on a diskless workstation instead of an Xstation. The disk and memory are no longer local, so the data must be transferred over the network. This becomes the bottleneck.

The conclusion is that, in terms of performance, the Xstation is the right choice for an I/O-intensive workload. The fact that data must not be shipped over the network for the Xstation is the reason for the striking difference in the results of this experiment.

CPU-Intensive Workload: A CPU workload running on an Xstation can be faster than on a diskless workstation, if there are few (about no more than three) Xstations. If there are more Xstations, then diskless workstations will usually perform better.

For a CPU-bound application, the sensible approach is to spread the workload among multiple CPUs. Diskless workstations are the better solution for these types of applications.

If the program can be cached in memory, multiple simulations make no difference to the workload of the file server. When each program has its own dedicated CPU, as it is the case with a diskless workstation, the execution time remains almost constant.

Software-Development Workload: Software-development environments mostly use compilers, editing tools, and operating system commands.

For typical software development, 63 percent of the elapsed CPU time is charged to the user, 30 percent is idle time, 4 percent is system time and 3 percent is I/O wait time.

The optimum configuration depends upon the number of system users. If there will be more than four working at the same time, then diskless workstations are more attractive because the workload is clearly CPU intensive, and this is where the diskless workstation performs best.

7.7.5 Conclusion

The fundamental difference between Xstations and diskless systems is that Xstations are high-function terminals and diskless systems are less-configurable workstations with no local storage. Dataless systems use a local disk for paging and local storage but require a server for all the other diskless functions. The diskless systems always execute a whole operating system, whereas Xstations have capabilities limited to the functions of an X Windows server.

Both systems have a close dependency on the LAN throughput. In the Xstation, it is mostly X Protocol that is traveling on the LAN. Diskless workstations need the LAN for every disk access (including paging), and the number of accesses is an important factor for a successful operation.

As a rule, consider Xstations when:

- Many files are accessed
- Heavy paging occurs

Consider diskless workstations when:

- User application(s) requires a dedicated CPU
- Upgrade options are important

7.7.6 References

The IBM Xstation Handbook, GG24-3695

AIXpert August 1992

Document N. OZSQ578570: Sizing a diskless environment

AIX V4 NFS/NIS/Diskless Class Material

7.8 HACMP Server Sizing

The purpose of this part is not to describe in detail the internals of high availability cluster multi-processing (HACMP). We assume that the reader has a basic knowledge of this product.

We will first review the major high-availability notions and will then briefly review the HACMP product before giving a few guidelines on how to size these servers.

7.8.1 High Availability Environment

There are some terms we need to define in order to understand how a high availability (HA) environment works.

Availability: It is a measure of the time a system has to be up and running, in comparison to the total time. It is often expressed as a percentage.

Outage: It is a planned or unplanned loss of application service.

Cluster: A set of independent systems connected over a network. In most cases, a few external storage devices are shared among those systems.

Node: Each system that is part of the cluster.

Storage: For HA, storage is defined as all persistent forms of storage, such as disk.

Resources: Everything that is considered critical for the provided service and can be shared between the nodes. This may include one or more of the following logical devices: disks, volume groups, file systems, network addresses, or applications.

Single Point of Failure: If the failure of any of the cluster components results in the unavailability of a service to the end users, the component is called a single point of failure (SPOF). Some possible SPOFs for a cluster are:

- Individual processor or node in the cluster.
- Disks used to store application executable or data.
- Adapters, controllers, and cables used to connect the nodes to the disks.
- Network adapters attached to each node.
- The network backbones over which the users are accessing the cluster nodes.
- Asynchronous adapters.
- Application programs.

All these SPOFs can be avoided in two ways: manual or automated procedures. As for the manual procedures, the time to recover from any failure in a cluster

could range between a few minutes to several hours depending on the type of failure (excluding diagnostics time). On the other hand, for automated procedures, such as HACMP, the time to recover from failure in a cluster could range between a few seconds and a few minutes. (This time does not take into account the application recovery time.)

Availability is the main goal to achieve. It is associated with the working cost of the system, and it seeks a reliable hardware and software performance, providing a reasonable level of desired availability.

We can classify availability in four major levels:

1. **Base Availability**

   This is achieved with a simple system and basic system management software. For some applications, it is a sufficient level of availability, but you have to plan for system outages in the normal cycle of system operation (including the expected recovery times from failures).

   Reliable hardware and built-in software can provide availability improvement while achieving a good cost/reliability trade off. Most modern computer systems are engineered with features that provide various fundamental levels of availability. The RS/6000 and AIX are rich in features that improve system availability, and many of these features are invisible to users and only become obvious during system recovery.

   Here are several tools for improved availability packaged with AIX or the RS/6000:

   - System Management Interface Tool (SMIT)
   - Logical Volume Manager (LVM)
   - Journaled File System (JFS)
   - Dynamic Kernel
   - System Resource Controller (SRC)
   - Configuration Manager
   - Built-In Error Detection and Correction
   - Battery Backup System (only on rack-mounted RS/6000)
   - Power conditioning (into the normal power supply)
   - Hot-Pluggable Disk Drives (on certain RS/6000s)

2. **Improved Availability**

   A greater robustness can be provided with the addition of technology or resources to one or more system components. Disk mirroring, the use of an uninterruptible power supply (UPS), redundant components, data journaling, checksumming, hot-pluggable disks, RAIDs or disk sharing can be used to help overcome some system failures.

3. **High Availability**

   High availability attempts to provide continuous service, minimizing the causes of failure and the recovery time needed when failures occur. HA requires a large degree of redundancy in the system components so the system operation can be continuous and protected from a failure. The main objective of HA is to eliminate all single points of failure by having redundant components or systems and management technology to automate transferring the services to those redundant components or systems in case of failure. It is crucial to ensure that the recovery time from any unplanned outage is minimal. HA is part of the HACMP product for the RS/6000 family. It is also referred as fault-resilient computing.

4. **Continuous Availability (CA) or Fault Tolerance**

   This level of availability tries to provide 100 percent availability to the end users by providing both redundancy in components and the ability to perform all error recoveries and change processes online. With a CA scenario, clusters will detect and recover from failures of disk, I/O adapters and cables, networks, network adapters, and processors. Such a cluster provides redundancy by transferring control of resources from one set of nodes to other cluster nodes. Planned outages may occur, but they should not be apparent to the end users.

A cluster is made up of many components. The cost of the outage is the primary focus of HACMP. If your environment can tolerate a maintenance outage, you do not need HACMP or any other software product to meet your availability needs. But if your requirement is for a outage smaller than a minute, you should consider a solution such as HACMP.

7.8.2 HACMP Product

High availability cluster multi-processing (HACMP) is a software whose goal is to minimize the effects of a failure for the end users. It involves some special hardware requirements besides the HACMP software, depending on the application, environment and security level you want to implement on your site.

The High Availability Cluster Multi-Processing/6000 software has two different subsystems:

- **High Availability Subsystem**

  It provides a highly available environment in a cluster of RS/6000s. The subsystem automatically reconfigures replicated resources in case of hardware or software failures or outages. Applications and users that cannot experience interruptions are automatically restarted on another machine in the cluster. The clusters running HACMP are able to detect and recover from failures of disk, I/O adapters and cable, network, or processor. If the proper hardware is installed and required procedures have been implemented, the subsystem can transfer the control of a failed component to one in place within a node, or even, in case of a full node failure, from the failed node to another available cluster node.

- **Concurrent Resource Manager (CRM)**

  This is also called the loosely coupled multi-processing services, and it enables up to eight RS/6000 servers to concurrently access the same data and run the same application. It also provides failure protection for all nodes. It enables you to spread a workload across several RS/6000 servers inside the same cluster, thus sharing disk and/or CPU resources.

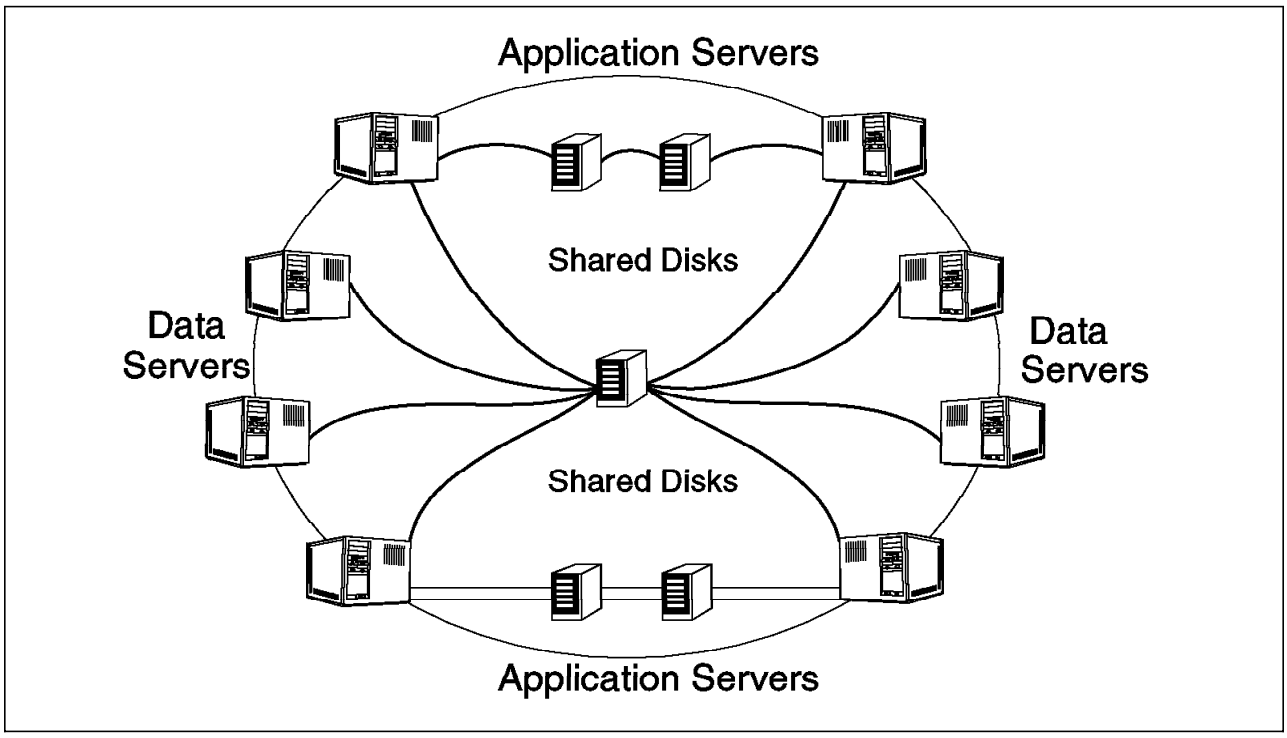


Figure 85. HACMP Cluster Configuration Example

An HACMP cluster consists of up to eight nodes, one or more strings of shared storage, a series of network interfaces on one or more networks, and the HACMP software. The storage devices are physically cabled to two or more nodes.

HACMP supports the following disk devices: SCSI-2 differential disks, SCSI-2 F/W differential disks, RAID subsystems, and SSA subsystems. Refer to Table 9 on page 72 for information about HACMP-supported disks or to 4.3, “Storage” on page 60 for detailed information on storage.

Depending on what happens to a given resource when the node that owns that resource fails, resources are divided into three classes:

1. **Cascading resources**

   A takeover priority is assigned to each configured cluster resource on a per-node basis. In the event of a takeover, the node with the highest priority acquires the resource. If the node with the highest priority is unavailable, the node with the next-highest priority takes the resource. When a node returns from a failed condition or when a node with a higher priority joins the cluster, it takes the resource.

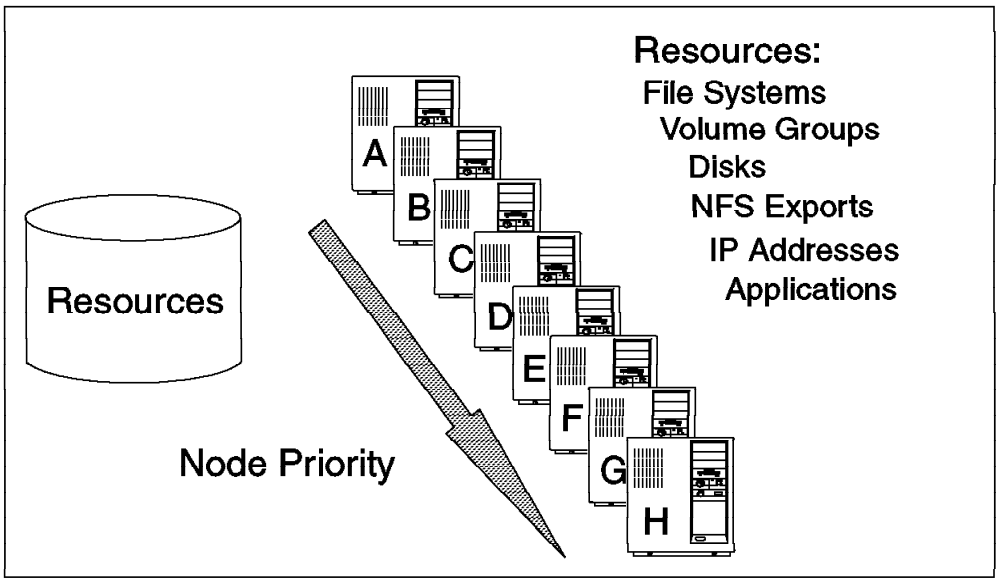


Figure 86. Cascading Resources

2. **Rotating resources**

   Each resource rotates among all the nodes defined in the cluster. The node with the highest priority takes the resource and holds it until a failure or a planned action makes it leave the cluster. No takeover has to be done when a node rejoins the cluster after a failure.

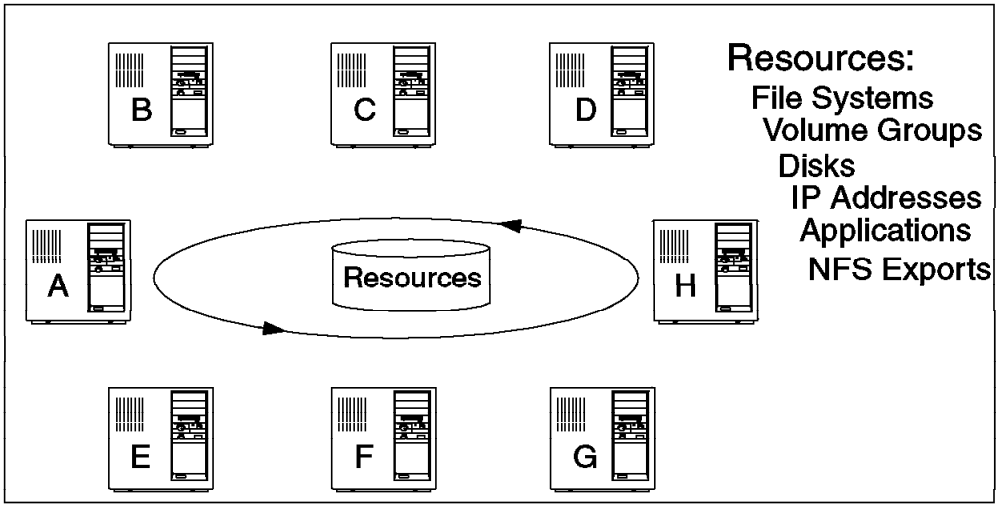


Figure 87. Rotating Resources

3. **Concurrent-access resources**

   Resources may be shared simultaneously by multiple nodes. There is no priority among nodes. The concurrent resources are limited to volume groups with raw logical volumes and raw disks.

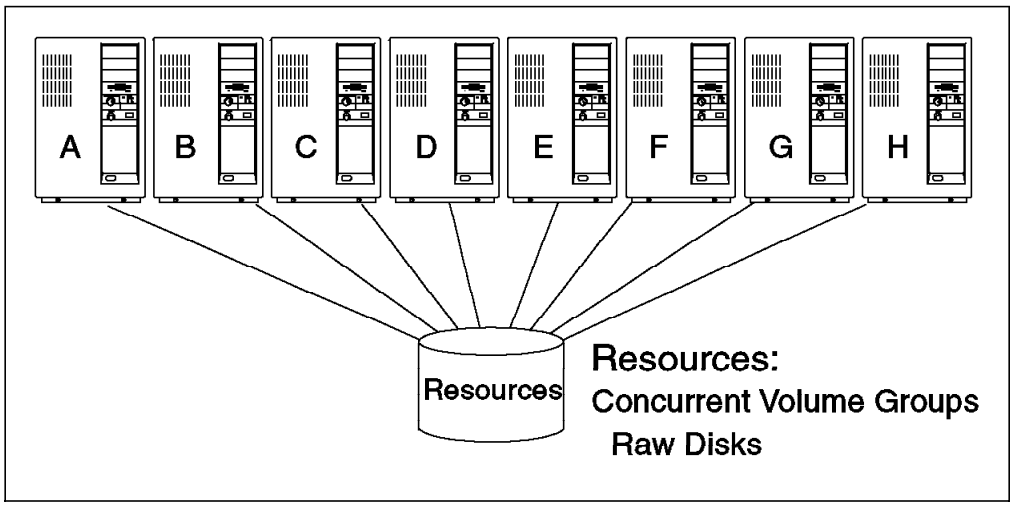


Figure 88. Concurrent-Access Resources

There can be several cluster configurations, depending on the way you want to manage your resources:

- Hot Standby
- Rotating Standby
- Mutual Takeover
- Concurrent Access

For more information on the HACMP product, please refer either to your IBM sales representative or to the following redbooks (that may also be consulted on `http://www.redbooks.ibm.com`):

- *An HACMP Cookbook* (SG24-4553)
- *HACMP/6000 Customization Examples* (SG24-4498)
- *Implementing High Availability on RISC/6000 SP* (SG24-4742)
- *High Availability on the RISC System/6000 Family* (SG24-4551)
- *Backup, Recovery and Availability with DB2 Parallel Edition on RISC/6000 SP* (SG24-4695)

7.8.3 General Sizing Considerations

HACMP is designed especially for the RS/6000 server family. As for sizing, there are three main topics when dealing with HACMP:

- Processor
- Storage Devices
- Network

Processor: Planning considerations on which RS/6000 to choose for an HACMP environment are closely related to the application that will run, the number of users who will access the resources, the database software, and the size of the database.

As a starting point, the HACMP servers should be sized considering the total requirements of the application. The next step is to find out which availability level is required for your specific application. With these two elements, you can choose the most appropriate cluster configuration between hot standby, rotating standby, mutual takeover or concurrent access.

To size nodes, you have to think about the minimum performance of the solution during and after a failure, knowing that the surviving nodes have to add the workload of the failed one to their own workloads.

In an HA environment, avoiding single points of failure is the main effort. Thus, you have to ensure that all the SPOFs are covered for the availability level you need. To achieve that, enough free slots are needed to support network adapters and disk I/O adapters for redundancies, according to the cluster configuration needs. At least two network adapters are needed for an IP takeover to take place, and the number can grow up to seven standby adapters in a node. In the same manner, two (or more) disk adapters are recommended to support the cluster storage configuration and its anticipated growth requirements.

The number of free slots for a particular availability level will help determine the RS/6000 model you need for your application.

The minimum requirements of the HACMP software are:

- 32 MB RAM memory
- 15 MB of disk storage
- 5 MB additional for concurrent access tables

It is suggested to always configure more than the minimum requirement.

Storage Devices: Sizing the storage devices is entirely related to the application requirements. The disk capacity depends on the database or application parameters and on the level of availability needed in the configuration (like mirroring or RAID). The disk arrays supported by HACMP include:

- SCSI differential disks, both narrow and wide SCSI-2 arrays. This technology accepts up to four nodes connected to the array, provided that the total number of devices (disks + adapters) is not more than seven for narrow SCSI or 15 for wide SCSI and AIX V4.
- RAID arrays, with SCSI or SSA disks attached.
- Serial link disk arrays. These arrays accept up to four nodes connected to the array on the cluster.
- Serial storage architecture (SSA) disk subsystem. Connection to up to eight nodes is allowed in this technology.

Depending on the availability level required, these storage subsystems may need redundancy. The workload attributed to each array should be balanced depending on data types and throughput capacity of the array. Refer to 4.3, “Storage” on page 60 for those considerations.

The choice of a particular technology depends on solution requirements such as: inherent availability, functionality, performance, and cost.

Network: A very important part of any cluster is the network over which the cluster nodes are connected. There are several factors that affect your choice of a network for your cluster:

- Price
- Performance

- Reliability or bandwidth
- Cluster event detection rate
- Support of IP and hardware address takeover

When dealing with HACMP, the networks are divided into two types:

- Public

  It connects two to eight nodes in the cluster and allows all the clients of the systems to have access to these nodes. Ethernet, token ring, FDDI, ATM, and SLIP are considered public networks.

- Private

  It provides point-to-point communication between two nodes; it does not allow any client access.

Every cluster must have one or more TCP/IP networks (public), and at least one non-TCP/IP network (private). The non-TCP/IP network is used to send keep-alive packets (also called heartbeats).

An Ethernet network should be the best option for the public network if the collision domain is kept small or when you do not need the reliability provided by token rings or FDDI. Ethernet networks have a faster HACMP error detection (six seconds) than token rings, and they support IP and hardware address takeover.

If bandwidth and reliability are required, the options are token ring and FDDI. The error detection on token ring networks is approximately 12 seconds and on FDDI, around six seconds. But hardware address takeover is not supported on FDDI (only IP address takeover), and the number of necessary slots is greater and the price higher for FDDI. For detailed characteristics of the adapters and network performance, refer to 4.5, “LAN / WAN Adapters” on page 90.

Another option is the ATM technology (only on MCA-based servers for now), as it improves bandwidth (up to 155 Mb/s) and brings additional benefits such as: synchronous transmission, improving the heartbeat transmission; predictable response time; and switching technology with wider range of connectivity.

For the private network (also called serial network), the options are RS-232 serial link and target mode SCSI. The most-used link is the RS-232 cable. However, if you are using SCSI differential disks as shared storage, you can also configure a target mode SCSI network to provide as many keep-alive paths as possible between nodes.

7.8.3.1 HACMP and SP Considerations

The RS/6000 SP hardware has been designed for availability (among other functions). Redundancy between its hardware components and special elements such as the high performance switch improve system availability. Multiple HACMP clusters can be installed on a single RS/6000 SP system, and a cluster can have nodes in a different RS/6000 SP rack.

A few considerations should be made to properly configure an SP in an HACMP environment:

- SP node type

  You have to check the amount of free slots for the application and for the availability-level requirements, as explained before. Some nodes only have

four slots available, and that limits their configuration possibilities. In an HACMP environment, you will need at least two network adapters and an SP switch for internal communication between nodes. So, if you choose a four-slot node, you only have one slot left for the storage arrays, without redundancy on the storage side, which is not advisable.

- SP internal resources

  Considerations about memory and internal disks on SP are similar to those of SMP or uniprocessor models. Refer to 5.3.5, "Sizing and Configuring an SP System" on page 129 for detailed information about SP configuration.

7.8.3.2 HAGEO

The High Availability Geographic (HAGEO) cluster is a high-availability solution for two systems at remote sites. It is the best solution for disaster recovery because of:

- Real-time data and file mirroring.
- Remote hot backup.
- Auto takeover of failed system.
- Remote mutual takeover if either system fails.
- Remote system recovery, resynchronization and reintegration of the failed system.

The HAGEO requirements are similar to the HACMP ones, for the public and private networks. HAGEO uses geographical networks to monitor the state of the networks, networks adapters, and the clusters at the other site.

Most of the TCP/IP WAN networks are supported by HAGEO, such as: Frame Relay, ATM, DS1 (1.544 Mb/s transmission rate and contains 24 circuits), DS3 (44.736 Mb/s transmission rate and contains 672 circuits), OC3 (155.52 Mb/s transmission rate), ISDN, SMDS (Bellcore, 44 Mbit service), and others. The network is configured as two separate networks:

- Geo-primary, which carries data and heartbeats.
- Geo-secondary, which carries heartbeats and cluster messages.

A non-TCP/IP network provides an alternate communication path for monitoring.

Only the geographical network adds new parameters in comparison to HACMP sizing. The network choice will depend on the specific application requirement.

7.8.4 References

High Availability on the RISC System/6000 Family, SG24-4551

An HACMP Cookbook, SG24-4553

HACMP/6000 Customization Examples, SG24-4498

High Availability Cluster Multi-Processing 4.1 for AIX. Planning Guide, SC23-2768

High Availability Cluster Multi-Processing 4.1 for AIX. Locking Applications, SC23-2772

http://w3.austin.ibm.com/rsdeliv/pres/preview/hapresc.htm

http://www.austin.ibm.com/software/Apps/hacmp.html

http://www.rs6000.ibm.com/resource/aix_resource/Pubs/aixcellence/highavl.htm

7.9 Multimedia Server Sizing

Multimedia is a widely used term today. To understand the meaning of this term, we will give a short description of multimedia services.

Multimedia is a presentation of information using video, audio, text, and graphics to allow the communication interface between the computer and the users to be more natural. As a result, the term multimedia is used to define a class of computing in which multiple information data types, such as text, graphics, images, sound, animation, and video, are combined to improve the quality and efficiency of communication.

Multimedia is a set of enabling technologies that are combining two distinct devices, the computer and the television, into single technological equivalents.

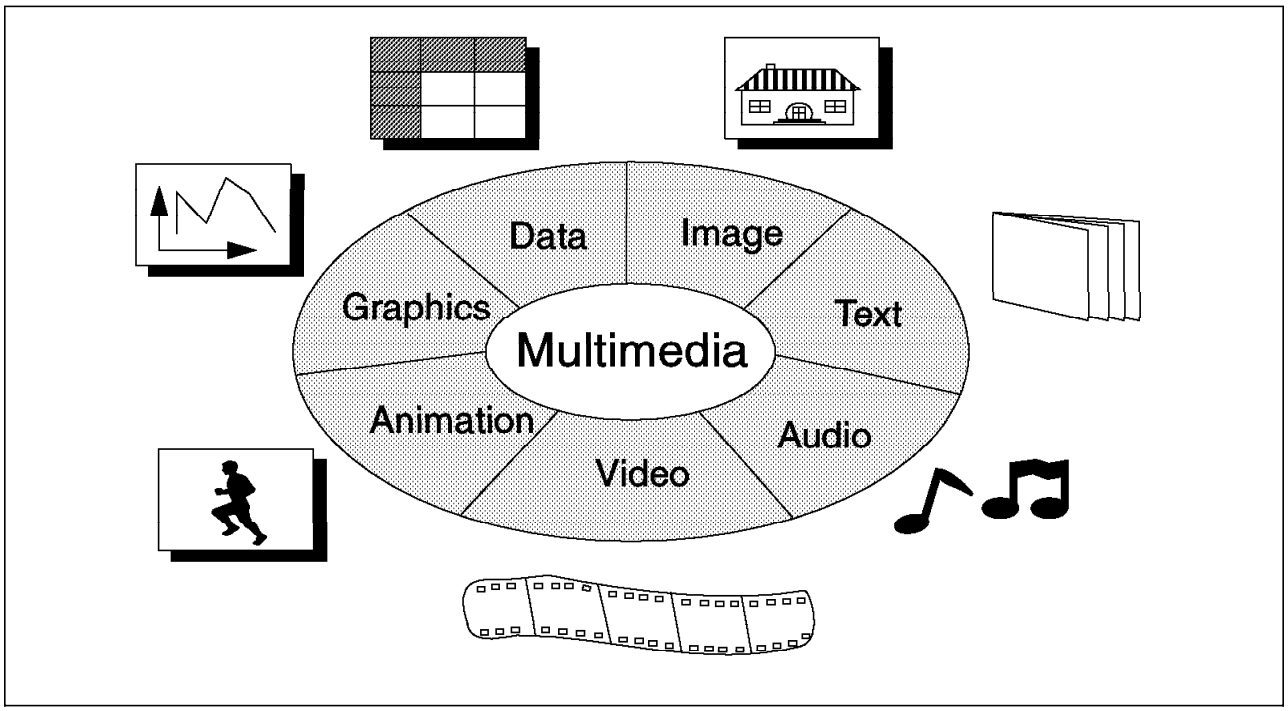


Figure 89. Multimedia Environment

7.9.1 Multimedia Environment

The multimedia environment consists of the following components:

Server: The server provides large storage facilities to store multimedia assets and controls the way application programs and multimedia tools access the data.

For large applications, a distributed computing environment is better. In such an environment, the servers store, catalog, and manage the multimedia data for the enterprise. They deliver audio/video data streams to clients. The request must be serviced transparently.

In the multimedia environment, the server needs to control very large volumes of data streams. A stream is simply multimedia-compounded data in a session, flowing over the network connection at specified rates to support the requirements of a multimedia application.

The server also controls the media that store the data and the network communication that distributes the data to and from the users in real time.

Network: The network is an important component of the multimedia environment. It has to deliver uninterrupted multimedia data streams to the clients.

Multimedia objects, such as audio and video, are time based, and data streams must be delivered at a constant rate in order to preserve human perception and application quality.

The objects must be sent in a controlled and synchronized fashion. That is why the network must guarantee an optimum bandwidth for multimedia traffic. There is a large demand on the network resources for data transfer. If the network cannot guarantee the bandwidth demand, the user experiences pauses in audio and video, and the multimedia quality suffers.

Multimedia server applications start as soon as the first part of the file is received. The client application starts to run and display video or plays audio, whatever the application is made for.

With the improvements in network technologies, one can build a distributed multimedia environment. Technologies such as Fast Ethernet, token ring, fiber distributed data interface (FDDI), asynchronous transfer mode (ATM), intelligent wiring hubs, bridges, routers, and concentrators protect your local area network investment.

Clients: The clients perform an important role by presenting multimedia to the user. To support multimedia data streams, clients need multimedia applications and special hardware.

Several clients can make simultaneous requests for the same or different data, located on the same or different storage devices. Without good network planning, these requests can increase the potential for server resource contention and data stream interruptions. Clients with different operating systems can all request multimedia support at the same time over the same network.

Storage: Multimedia allows you to digitize and store video, audio, text, and graphics information in databases. A digital video or high-quality audio file requires large storage capacity. For example, you need about 10 MB per minute for a VHS-quality movie.

The multimedia server provides the storage solution to the environment, with a centralized storage. It provides management and control that is currently used with other types of data, such as accounting, inventory, and customer records.

With multimedia solutions, you can address the problems associated with storing, managing, distributing, protecting, and using digital audio and video information. Improvements can be achieved such as the following:

- Multiple copies of source data can be eliminated.

- Compressed digital audio and video can be distributed over networks for easier distribution.
- Multiple users can access the data at the same time.
- Storage requirements for desktop clients can be reduced.
- Large block file transfers on the network can be reduced because data is streamed to the clients.
- Replicated data provides a backup to prevent loss or contamination.
- Data does not degrade or lose quality during replication.
- Data can be modified and reused more easily.

Data Management: Multimedia servers use a special file system optimized for audio and video data format delivery. This high-performance file system gives the multimedia file server great achievements, such as:

- Supports large files in file systems.
- Each file system can support up to 64,000 disks (up to 256 terabytes of data).
- Supports reading large blocks of data (more than 256 K). Reading large blocks of data reduces the number of read requests the server must handle.
- Uses disk bandwidth efficiently and provides predictable disk throughput.
- Calibrates disks automatically to effectively use the available bandwidth when a new disk is added to the system.
- Uses deadline scheduling to manage input and output for real time delivery.
- Scales to support a range of independently controllable streams.
- Balances load regardless of file access patterns.
- Remains operational during hardware failure and reconsideration.

The multimedia assets are consolidated and distributed in the file systems for storage and retrieval using a striping method. Striping spreads a file across multiple disks in a file system. It increases the bandwidth of the file system, balances the load on the disks using the combined throughput of all disks, and allows concurrent reads from the file system.

Additionally, multimedia servers use replication of files. This ensures that the system can continue to run and deliver data to the users in case of a failure. Replication puts multiple copies of data in the file systems.

The multimedia high-performance file system works in parallel with the well-known journaled file system (JFS) and uses some of the JFS functions.

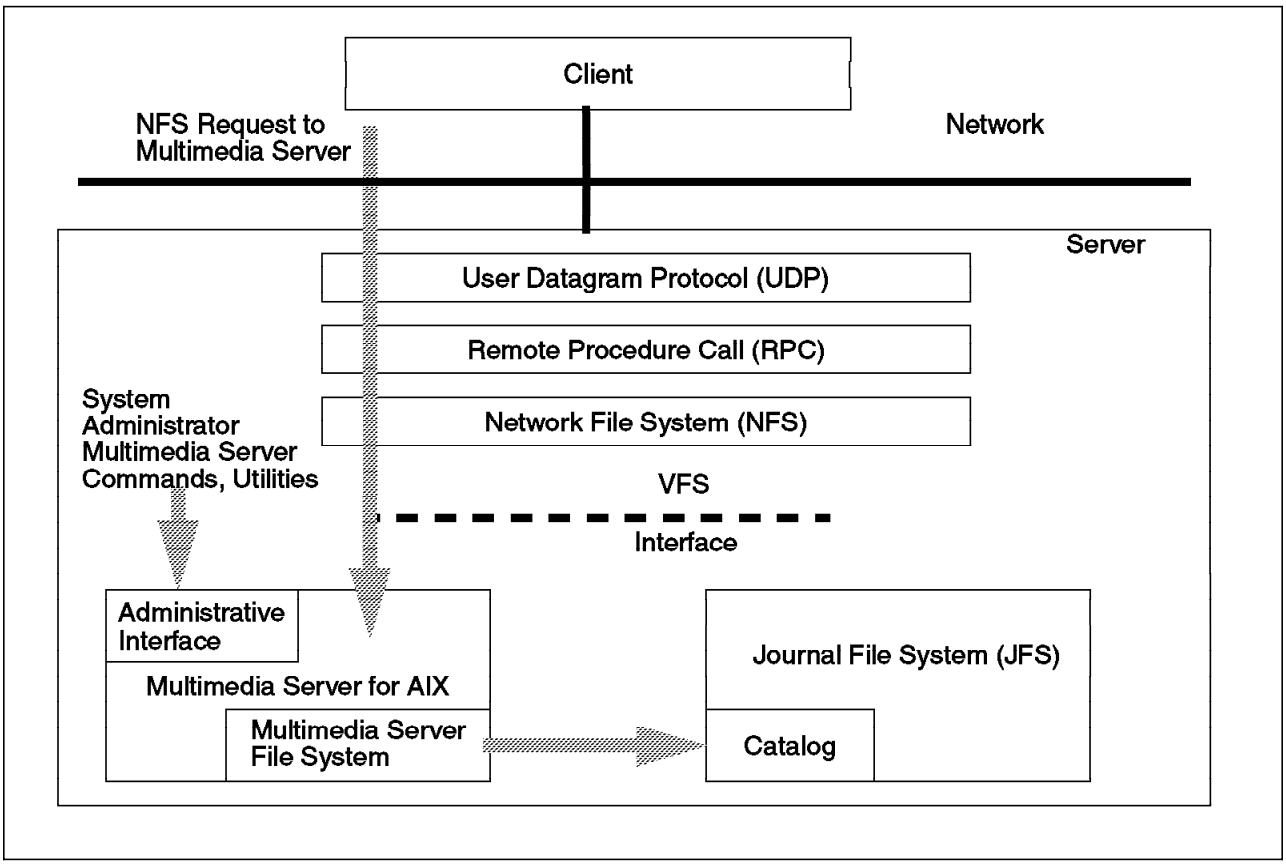


Figure 90. Multimedia Requests Relationship

7.9.2 General Sizing Considerations

Sizing for a multimedia environment consists of two parts: server(s) and clients.

Server Considerations: The multimedia software is not supported on all RS/6000 platforms. In general, it is supported by workgroup and enterprise servers.

When you select a server you should consider the following:

- How much data do you want to store? You have to size your disk requirements accordingly. An IBM SCSI-2 Fast/Wide disk drive can hold approximately two hours per GB of data content, assuming that it is encoded at 1,200,000 bits per second (storage need is about 150 KB per second). The RS/6000 storage family has solutions such as SSA disk arrays and RAID disk arrays. Refer to 4.3, “Storage” on page 60 before configuring the server and check the multimedia web page (`http://www.rs6000.ibm.com/resource/`) for information on supported disks and systems.
- Which type of network are you going to use? You need to choose the correct network for your hardware and client requirements. Depending on the platform selected, multimedia server configurations have the capability to support up to 62 simultaneous streams of 150 KB of multimedia data. Basically, the options for the network are:
  - Ethernet

    The Ethernet solution does not allow you to connect many users, because Ethernet is a probabilistic transmission media and cannot

guarantee the bandwidth for large amount of clients. A multimedia client typically reads about 150 KB per second. This means, for an Ethernet network, you can connect up to five clients per segment. Approximately 28 clients are allowed for a Fast Ethernet. The number of adapters allowed in the network is determined by the server. Refer to 4.5, “LAN / WAN Adapters” on page 90 for detailed information about the adapters.

- Token Ring

  The token ring solution is a dependent transmission media. To guarantee 150 KB per second bandwidth for the clients, you should not connect more than eight clients per segment. The number of segments and adapters allowed is determined by the server.

- FDDI

  With FDDI networks, the same concept as for token ring is applicable. You should allow up to 32 clients per segment.

- ATM

  ATM is an isosyncronic transmission media. It may be the best option for the multimedia environment. With ATM, you should allow up to 46 active clients per segment.

- What kind of display adapter you need? Although ASCII terminals are supported for the multimedia server, it is highly recommended to use a graphics display. For most applications, an S15 graphics adapter is enough to play motion video and other applications. Optionally, IBM offers a video adapter for both the RS/6000 Micro Channel and PCI bus-based systems. This adapter is a multipurpose video adapter called the Ultimedia Video I/O adapter. It supports real-time video monitoring on the desktop, video output of YUV data as NTSC or PAL (TV image formats) and a single-frame image capture. You can add a video compression guest card for real-time video capture with MJPEG compression and for compressed JPEG stream output to video support. Another option is the Video Capture Enhancement adapter for systems with PCI bus and S15 graphics adapter. This combination supports video monitoring from standard PAL, NTSC, or SECAM sources (such as video cameras, VCRs, laser disk players) and also supports single-frame capture.
- How much memory is needed? The absolute minimum is 16 MB, but it is strongly recommended to configure more memory. The amount depends on the applications that will be running. Configurations with 64 MB and 128 MB support movie applications.

Client Considerations: Multimedia clients must have the capability and capacity for receiving and playing multimedia streams. For the RS/6000 entry workstations, most of the considerations mentioned for the server are applicable for the client side.

The difference is in the amount of memory required and the storage capacity. You need a minimum of 16 MB of RAM (32 MB is better) and a disk for the AIX and Ultimedia software. You also need a CD-ROM drive, video and audio adapters, and speakers.

7.9.3 Configuration Examples

The following configuration descriptions can be used to get a general idea of the important components in a multimedia environment.

Entry Level (up to 25 Mb/s throughput): Maximum 20 users or streams at 1.2 Mb/s per stream. A small configuration for new enterprise.

Description	Quantity
RS/6000 Model 7024-E20 with 128 MB memory and 6 GB disk storage	1
Single-port Ethernet, PCI (maximum 5 streams per adapter)	4
AIX Version 4.1.4, Server Code	1
IBM Multimedia Server for AIX: 25 Mb/s video streaming capacity	1

Figure 91. Configuration for a Small Multimedia Server

This configuration would be good for a small user community. It holds approximately eight hours of data content.

Midrange (up to 50 Mb/s throughput): Maximum of 42 users or streams at 1.2 Mb/s per stream. This configuration support approximately 20 hours of data content, depending on the type of multimedia services you plan to provide.

Description	Quantity
RS/6000 Model 7025-F30 with 128 MB memory, 15 GB disk storage, 1 SCSI-2 Fast/Wide adapter	1
Token ring adapter (maximum eight streams per adapter)	6
AIX Version 4.1.4 server code	1
IBM Multimedia Server for AIX: 50 Mb/s video streaming capacity	1

Figure 92. Configuration for a Midrange Multimedia Server

High End (up to 75 Mb/s throughput): Maximum 62 users or streams at 1.2 Mb/s per stream. This configuration holds 32 hours of data content, depending on the type of multimedia services you plan to provide.

Description	Quantity
RS/6000 Model 7012-G40 with 256 MB memory, 20 GB disk space, 3 SCSI-2 Fast/Wide adapter/A	1
Additional 2-Way processor card	1
IBM RS/6000 Model G02 expansion cabinet	2
Fiber distributed data interface	1
AIX Version 4.1.4 server code	1
IBM Multimedia Server for AIX: 75 Mb/s video streaming capacity	1

Figure 93. Configuration for a High-End Multimedia Server

7.9.4 Performance

The multimedia server performance depends on several hardware and software components. Variables affecting hardware performance are:

- Disk performance
- Disk type and configuration
- Disk controller performance
- Server system performance (processor and memory)
- Server network adapter performance
- Network type and configuration
- Client network adapter performance
- Client system performance (processor and memory)
- Application environment

The multimedia software is effected by:

- File and data storage management configuration
- Retrieval and archives functions
- Network file system (NFS) input and output

These components need to have the highest priority on the system, because they handle the real-time demands of the multimedia traffic. Refer to *IBM Multimedia for AIX, Guide and Reference* (SC24-5799) for information about tuning a multimedia server.

7.9.5 Conclusion

The multimedia market is growing. Applications for communication, training, education, customer services, audio/video archives, and broadcast video servers are common in multimedia environments. The integration of multimedia servers into the Internet and intranets brings a wide range of applications. Multimedia provides several tools to make it easy to use for the end user. It provides a robust application program interface for programmers, and lower level objects to directly control the hardware.

When planning a multimedia server configuration, you have to focus on two important aspects:

- Consider the best network technology you can afford, because it allows more users in a network segment.
- Consider new hard disk models with good performance and high capacity.

AIX with Ultimedia Services has some nice tools to monitor and tune your server for better performance.

7.9.6 References

RISC System/6000 Multimedia Environment: An AIX Ultimedia Services/6000 Overview, GG24-4254

IBM Multimedia Server for AIX. Guide and Reference Release 1, SC24-5799

IBM Multimedia Server for AIX. General Information Release 1, SC24-5798

http://www.rs6000.ibm.com/software/multimedia

http://www.rs6000.ibm.com/resource/

http://w3.austin.ibm.com/afs/austin/depts/rsdeliv/pres/preview/rsprmm2.htm

http://w3.austin.ibm.com/afs/austin/depts/anhs/public/docs/intranet.html

RISC System/6000 Technical Workstation and Graphics Marketing Guide, G520-7168

7.10 Web Server Sizing

The purpose of this section is to give guidelines on how to choose and size IBM RS/6000 Web server machines. The information in this document represents a set of guidelines that can be used to approximate the size of a server. This guide is only one of many resources available to assist people in developing IBM Web server solutions. This chapter will not discuss other aspects of Web servers, such as security and guidelines on how to choose Web server software.

7.10.1 Introduction

Since its introduction several decades ago, the Internet has undergone incredible growth. For many people, the Internet is an important aspect of daily life. It links abundant resources and information across the world and enables everybody to travel the Net (known as surfing) in a very simple way. It also introduces new ways to do business and makes online information accessible.

One of the Internet technologies that has been exploited very much is the World Wide Web. It enables us to see, search and post information across the world. By implementing some new technologies such as Java, it even enables Web documents or information to do interactive things.

Web technology is based on the Hypertext Transfer Protocol (HTTP). It is layered on top of TCP/IP in order to guarantee good data transfer. Usually, the machine that provides the HTTP service is called the Web server. This Web server can run on many platforms. On most UNIX machines, the process that provides HTTP service is called `httpd`. It usually runs as a UNIX daemon process and normally uses and listens to TCP port number 80.

Most of the existing Web servers run on UNIX machines because TCP/IP is integrated with the UNIX operating system and many tools to support the Web server are available as public domain or shareware software.

7.10.2 Sizing Information

IBM offers Web server solutions through the IBM RS/6000 Internet POWERsolutions program. This consists of several preconfigured IBM RS/6000 machines along with Web server software. Usually, before a customer chooses the Web server machine, he or she will ask several common questions, such as:

- How big an RS/6000 will be needed for a Web server?
- How many hits per day can an RS/6000 handle?
- What is the maximum number of clients that can be supported by an RS/6000?

Before we can answer these questions, we have to get information such as:

- Is the Web server going to be an Internet or Intranet server?
- What is the potential demand for access to this site?
- What it the speed of the connection to the Internet or Intranet?
- How many pages will be serving?
- What is the average file size of the pages?
- Will the Web server be generating data for access?

7.10.3 Sizing Factors

7.10.3.1 Target Environment

When sizing a Web server, the most important consideration is the size of the target audience.

Internet: Sizing a Web server for the Internet can be a very difficult task. The Internet includes millions of interconnected individuals who are navigating from one Web server to the next in search of information that has value to them. Sometimes, it is very hard to estimate how popular a site may become. Usually, for initial implementation, the Web server machine is chosen based on certain maximum accepted connections in acceptable response time. Later, based on the average statistical log, it can be decided to expand the system according to the growing usage. Therefore, if the growth factor is going to be considered, then the upgradability and scalability of the Web server machine should be considered, too.

Intranets: Intranets are private nets that use the same standards and protocols as the public Internet. Intranets are rapidly deploying internal Web sites as the new network-centric corporate computing platform. An Intranet Web site dissolves all departmental, geographic, and technical boundaries by creating a universal way to connect people to people and people to information.

Sizing a Web server for an Intranet is considerably easier than sizing one for the Internet. The total number of potential users can be determined more accurately by using the total number of employees in the relevant department or the entire company.

7.10.3.2 Network Bandwidth

In sizing the Web solution, it is important to understand the implications of the speed of the networking connection to the Web server. More often than not, many potential Web content providers are very focused on the vague hits-per-day quantity. The level of traffic that a particular Web server can support will be dependent on the server type, the content accessed on the server and the speed of the connection of the server to the Intranet or Internet environment.

An Internet service provider will deliver a connection of a defined speed. Three of the most common WAN speeds are: ISDN (128 Kb/s), T1 (1.544 Mb/s), and T3 (45 Mb/s). For an Intranet environment, common LAN speeds are 10 Mb/s (over Ethernet) and 100 Mb/s (over Fast Ethernet or FDDI). It is possible for the WAN or even the LAN to become the bottleneck of a high-performance Web site. This is especially common when non-Web traffic occupies the same network, degrading the site's performance. When this occurs, the network backbone must be scaled up to achieve maximum performance. Figure 94 shows the interrelationship between the average Web transaction size, the speed of the networking topology, and the maximum theoretical hits per second. To translate this into a number of hits for an approximately eight-hour peak usage period, multiply the hits per second by 28,800.

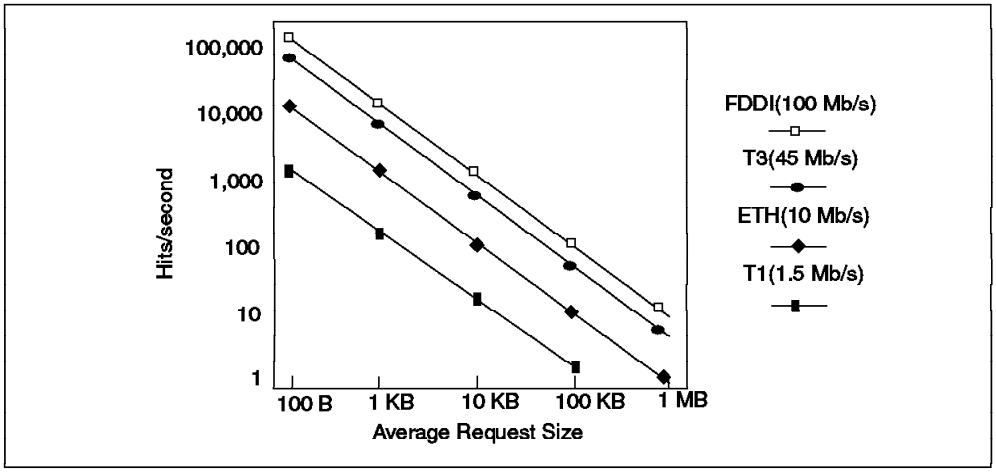


Figure 94. Relationship of Network Speed, Request Size and Maximum Hits

As the average Web transaction size increases, the maximum number of transactions decreases. Sites that plan on being mostly text-based will have average transaction sizes around 1 - 5 KB; most well-designed sites with a mix of text and graphics intended for access by modem users handle transactions of about 10 KB each; and sites with a substantial portion of multimedia content can exceed 100 KB per transaction.

7.10.3.3 Server Content

The content being served will, to a large degree, dictate the overall performance of the site. A Web site's content ranges from text to graphics to more complex multimedia file types such as video and audio. The type of content is closely related to the size and number of data transmissions, as text files are usually smaller. Graphics files are several times the size of text files, and multimedia data types are several times larger than graphics files.

It is not necessarily true that smaller files are better for performance because, if a Web server is required to serve large numbers of small files, the server performance may degrade. If the site is made up of large files, and most of the users are connected over low-bandwidth connections such as modems, the site's performance will be unacceptable to those users. The balance of the size and number of files required for a Web site must be considered as the appropriate server is chosen. The physical size of the Web content is important in determining the data storage requirements.

Pages can be static (they already exist on the server and are waiting to be requested) or dynamic (created on the fly by the Web server based upon the user's input, often in combination with the results of a database query). Most of the content that exists on the World Wide Web today is static (marketing-brochure type information, etc.), but the trend is toward interactive, dynamic Web sites. Sites that are integrated with other business applications or that are created for the purpose of doing business electronically are, by nature, dynamic and interactive.

Obviously, dynamically created pages require more server power, and any interaction with a database engine also requires additional server processing power.

7.10.3.4 User Interaction, Hits and Connections

Web sites can interact with users, usually by means of CGI script programs. The scripts capture user input and customize some aspect of what the user sees from that point on based upon the user's input. The user's input can also be used to formulate database queries that retrieve specific information the user has requested from external databases that may reside on the server machine or on an attached machine. This information is then displayed on a dynamically created Web page. The programming involved in capturing user input as well as the processing power required to perform or send and receive a database query will have an effect on Web server performance.

The complexity of a page determines how many connections (hits) are required of the server. A page that consists of a single HTML text file would require one hit. If that file also referenced three ″.GIF″ graphics files, four hits would be required to serve the page. If the page includes a user input area, additional connections would be required, and so on. The number of connections required to serve a single page adds up quickly. This can be seen on some Web pages that count how many connections were made.

7.10.3.5 Number of Clients

The number of simultaneous users of a site is very challenging to characterize. Unlike other types of client/server architectures, the weight of an individual client on the Web server is quite small and short lived. Connections to a Web server are traditionally stateless sessions that begin with an open from the client (a request for data); the server replies with data, and the session closes. Depending on the speed of the network connection, the size of the data requested and the server load, this session can last from less than a second to many seconds.

7.10.4 Web Server Performance

Web server performance measures several important areas that directly impact user experience and cost of ownership of the Internet solution. Usually, the following categories are used for the measurement:

- Response time

  This measures how long it takes the server to answer a client request. This is an important measurement to analyze, especially as the number of client connections increases and as the type of requests varies from static HTML to dynamic content creation using APIs and RDBMS. Usually, for a server to be fast, its average response time must be well under one second.

- Throughput

  Simply put, this measurement establishes the maximum amount of data the server can send through all open connections during a given amount of time. If the throughput is close to the bandwidth of the network (LAN or WAN), then the network is probably saturated.

- Connections per second

  This measures how many HTTP requests a server can establish, service and then close during a given period for a specific set of HTML files. To adequately capture a server's performance in this area, the server must be tested against small, medium, and large static and dynamic HTML files and applications. This metric also appears as hit per day. However, connections per second are more helpful in planning for peak loads.

- Errors per second

  This measurement identifies how many HTTP requests were not serviced or were dropped by a server. High error rates translate into an unreliable server that cannot handle the load at which the errors were generated. Ideally, no errors should occur.

7.10.5 Case Studies

In order to give an idea about the sizing of Web server machines, here are the results of some tests done on an IBM RS/6000 43P, 590, J30 and SP. All these tests are based on WebStone 1.0 or WebStone 1.1 benchmark criteria. For detailed information about WebStone benchmark, please see 6.8, “WebStone” on page 173.

7.10.5.1 Machine Configurations

The test configurations and a dynamic page content configuration were run on the following Web server hardware systems to look at the scaling properties of RS/6000 hardware in this application environment.

Hardware

1. RS/6000 43P-133 (7248-133) - Single PowerPC 604 Processor
   - 133 MHz PowerPC 604
   - 512 KB synchronous L2 cache
   - 128 MB memory
   - 2.2 GB disk
   - Integrated SCSI and Ethernet controllers
2. RS/6000 590 - POWER2 Architecture
   - 66 MHz POWER2

- 256 KB synchronous L2 cache
- 128 MB memory
- 2 GB disk
- Micro Channel Ethernet and FDDI controller

3. RS/6000 J30 - SMP PowerPC 601

   - 75 MHz, 4-way PowerPC 601
   - 32 KB synchronous L1 cache/processor
   - 1 MB synchronous L2 cache/processor
   - 128 MB memory
   - 4.5 GB disk
   - Micro Channel Ethernet and FDDI controller

4. RS/6000 SP - Parallel Processor

   - SP 66Mhz, two thin nodes (39H equivalent)
   - 2 MB L2 cache/node
   - 128 MB memory/node
   - 4 GB disk/node
   - Micro Channel FDDI controller

Operating System: AIX 4.1.4 + (PTF 443072)

Server Software: Netscape Commerce Server (NCS) Version 1.12

Tuning

- Maximum processes per user set to 256
- DNS reverse lookup off
- Netscape MaxProcs 200 (Uniprocessor, SMP), 128 (total on all SP nodes, for example, 64/node on a two-server configuration)

Four nodes on an RS/6000 SP1 drove the Web client processes. Each node (370 equivalent) was equipped with 128 MB RAM, 1 GB local disk and Micro Channel Ethernet and FDDI adapters. For the SP benchmarks, two 39H nodes and one 590 node were used as Web client drivers over FDDI.

7.10.5.2 Static Test Results

A major portion of the content on the Web is static. This includes both images and textual data. The CPU resources required to serve such data are minimal. The IBM RS/6000 product line has a large performance range from entry systems to highly parallel processing configurations, and this range in performance is observable in static-content WebStone benchmarks only when the correct processor and network topology combination are achieved. Figure 95 on page 251 illustrates the range of HTTP connection rates with Web servers from the entry 43P through four SP nodes used in parallel.

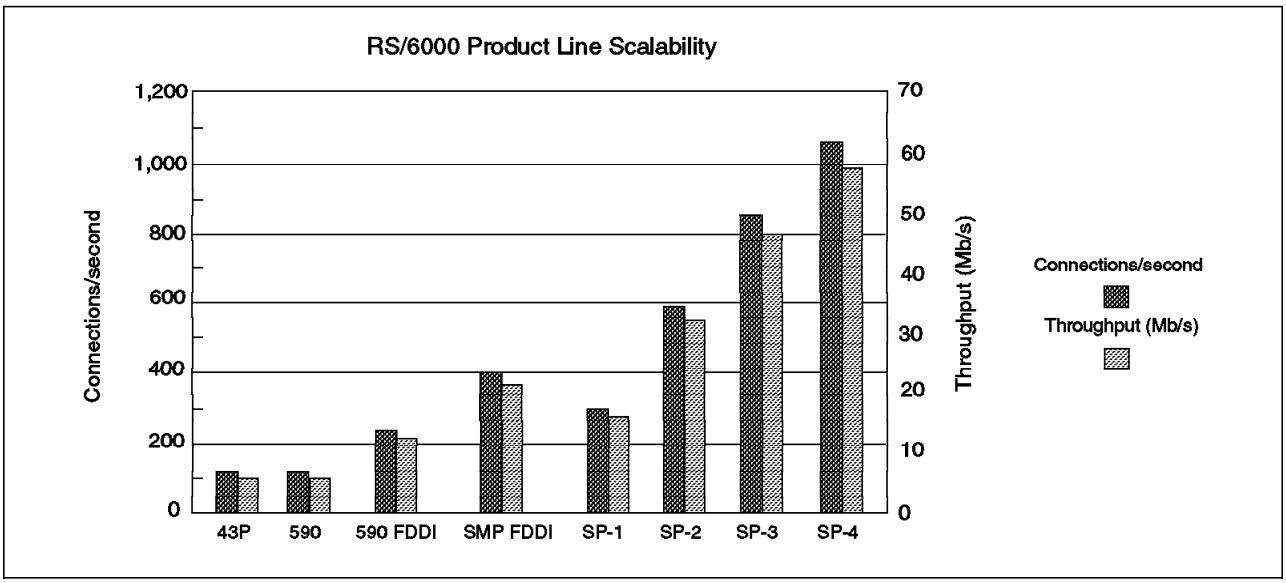


Figure 95. Static Workload Results

A typical HTTP connection consists of a client open, client request, server header and data response and connection shutdown. The average response size is approximately 7 KB. With FDDI, the bottleneck is at the CPU performance level for the midrange to high-end solutions.

As Figure 95 illustrates, a site that has static data and relatively small networking bandwidth capabilities needs no more than a properly configured entry uniprocessor system. At Ethernet networking speeds, even the 43P is constrained by the network and not the CPU. As sites expand and anticipate the need to accommodate T3 (45 Mb/s) speeds or for Intranet (LAN based) sites that require a database or other application on the Web server, a midrange uniprocessor, SMP or SP system is indicated.

7.10.5.3 Dynamic Test Results

When a Web server responds to users in a more dynamic way, it is a much stronger case for increased computing power at the server. Unfortunately, there has been very little work that characterizes the exact impact of various types of dynamic workloads on HTTP performance. With WebStone 1.1, the effects of dynamic workloads can be observed in a very simple way. Instead of returning an existing file, the Web server under test generates a file composed of random characters. Figure 96 on page 252 shows throughput in Mb/s over a range of RS/6000 hardware at both 25 percent and 50 percent dynamically generated page content.

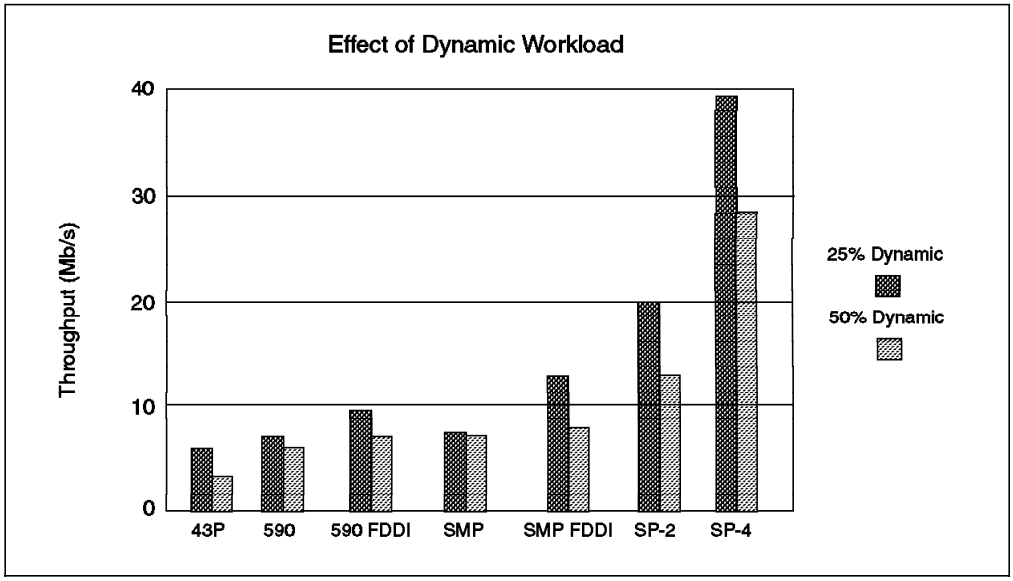


Figure 96. Dynamic Workload Results

The performance bottleneck in this benchmark shifts from the network speed to the processing speed of the Web server. In some configurations, there are still situations where the performance is network bound. For example, comparing the results for the 590 with Ethernet and FDDI, the output is network bound at 25 percent dynamic content generation and CPU-bound at 50 percent. The SMP system is heavily bound by the speed of the Ethernet network at 25 percent dynamic page content.

7.10.5.4 Client Workload Results

As the physical number of clients increases, overall response time from the server increases as well. Over the client range available for testing (WebStone 1 only produces statistics for up to 256 clients), the results show reasonable behavior in server latency with increasing number of clients. Figure 97 and Figure 98 on page 253 show the combined effects of dynamic workload and client quantity.

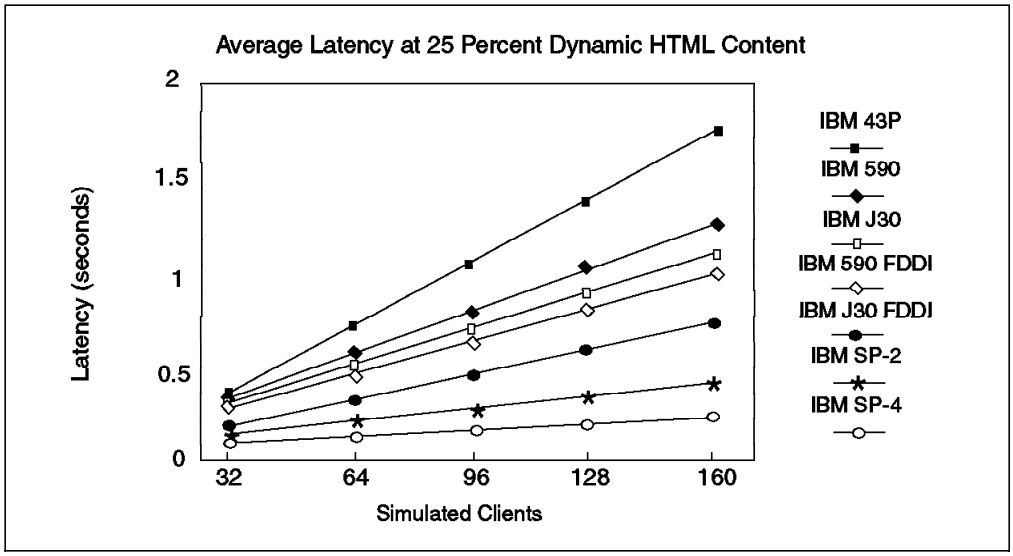


Figure 97. Server Latency at 25 Percent Dynamic HTML Content

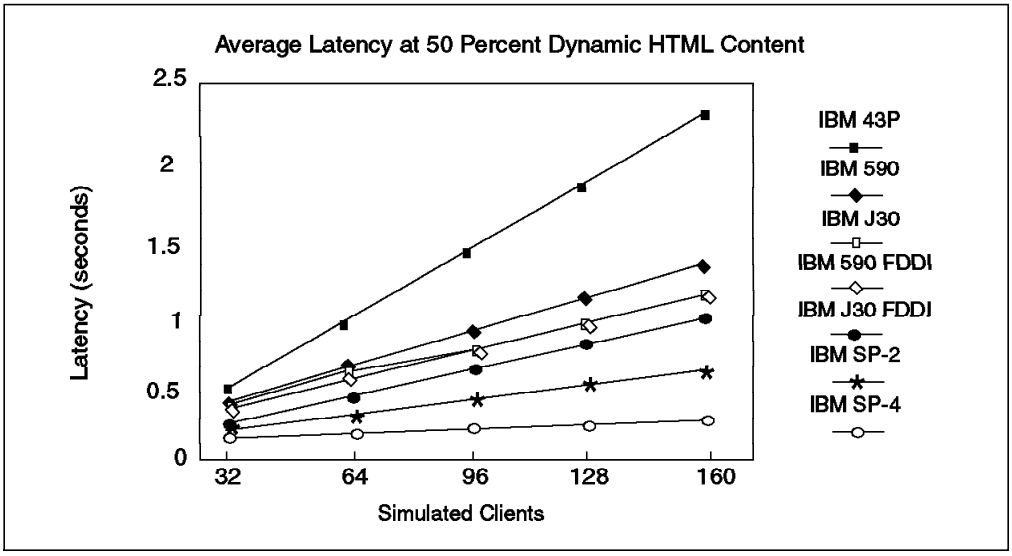


Figure 98. Server Latency at 50 Percent Dynamic HTML Content

Solutions that will perform best in a real-world, large client workload environment have lower slopes in these graphs. Looking at Figure 97 on page 252 and Figure 98, there is a close to linear relationship between number of clients and server latency over the client range displayed. The slope of these lines will be a strong indicator of the scalability of the solution. At 160 clients and 50 percent dynamic content, the 43P has an average latency of 2.3 seconds, approaching the upper range of acceptable delay. Higher-performance solutions have lower slopes, indicating a more graceful degradation under heavy workload. In general, these workloads reflect moderate traffic rates. Consider a vendor site with an average of 64 KB data per Web page (a mix of text, small icons, and a couple of small pictures on each page). If a typical user accesses 10 pages at the site, then over an eight-hour period, 32,000 users could be accommodated with a 43P at 50 percent dynamic content workload. In the test, a 590 over FDDI with 128 MB of RAM can serve data to more than 500 simulated simultaneous clients.

7.10.6 IBM RS/6000 Internet POWERsolutions Performance

As part of IBM's participation in the Internet solution, IBM has developed RS/6000 Internet POWERsolutions. Basically it is preconfigured RS/6000 hardware, from the entry level to the high expendability system, and Web server software. Table 27 on page 254 shows benchmark results of the different hardware. These benchmarks have been designed to simulate actual Web server workloads. However, real performance is determined by a number of factors that vary widely. Actual performance will be different. How different depends upon how different the real situation is from the environment simulated in the benchmarks. Therefore, you can choose any RS/6000 model that fits your particular situation. The measurement of these benchmarks is to get the number of peak connections per second. For this purpose, there are three types of workloads defined:

- Low - Little or no user interaction, primarily static text content
- Medium - Some user interaction, some dynamic page creation, mixed text and graphics content
- High - Highly interactive, dynamic page creation, multimedia content

RS/6000 Model	Low (connections/s)	Medium (connections/s)	High (connections/s)
43P 133 MHz 64 MB, 1.2 GB	206	124*	52*
43P 200 MHz 64 MB, 2.1 GB	320	192*	80*
F40 2-way 64 MB, 2.2 GB	352*	211*	88*
G40 2-way 64 MB, 2.2 GB	434	260*	109*
SP, 2 nodes 128 MB/node 4 GB/node	810	486*	203*

Table 27. Performance of Preconfigured RS/6000 Web Servers

Performance figures that are estimates are indicated by "*".

In general, more complex Web sites require more processor power. The figures in Table 27 are derived from actual measurements, as well as theoretical models, and are in the process of being further validated.

Much of the performance measurements done in the Web server business have been done in the realm of static, text-based Web servers, and thus, the first data column in the table is fairly accurate. As the complexity increases, so do the number and the variability of the parameters involved. Therefore, the expected performance in these scenarios is much less certain.

For consistency, the data represent connections/second using the Netscape FastTrack Server running with AIX 4.1.5. Higher connections/second measurements can be, and have been, achieved using AIX 4.2 and/or Zeus Web Server. Using AIX 4.2 with Netscape products yields a performance gain of about 30 percent on WebStone and SPECweb96 benchmarks because the executable code for the Netscape products is different for AIX 4.2 and takes advantage of hardware threading. The Zeus Server yields higher performance results in general, and it is now being used by most of our competitors to publish Web server performance figures. The impact of using AIX 4.2 with the Zeus Server is minimal because Zeus employs the same executable code in either environment.

7.10.7 Web Sizing Tools

There is a home page (in development/beta test) in IBM's internal network that provides an online Internet sizing tool. It is at:
`http://w3.austin.ibm.com/cgi-bin/WebSz1.pl`

This sizing tool can give a recommendation of RS/6000 machines for Web servers based on sizing requirements.

7.10.8 Recommendations and Conclusions

In general, Internet sites that are connected by T1 lines and Ethernet LAN-connected Intranet sites with largely static data are adequately served by an entry uniprocessor system with adequate disk storage for the content provided. It is important to have enough RAM to accommodate both the HTTP server processes (64 MB was adequate in the testing) and for file caching of page content that resides on disk.

Sites with high-bandwidth connections to the Internet and Intranet sites that can utilize FDDI will benefit from midrange and SMP solutions. Sites that will generate significant Web content in response to user actions or potential electronic transaction sites should consider such systems even if they are connected by T1 lines to the Internet or by Ethernet LAN to the Intranet.

The best possible scale-up with increasing client workload is accomplished through the use of IBM SP systems, or clusters of systems. This solution depends upon use of the Round Robin Domain Name Service to direct clients to multiple nodes running mirrors of the Web site. In addition, to get a linear scale-up, care must be taken to make sure that each node has an adequate dedicated connection to the Intra/Internet. In the benchmark results, four nodes of the SP would be well-suited to fill the full bandwidth of a T3 line for distribution of static Web content. An SP system brings strengths of better management and faster intranode communication of shared data. For the Intranet, the common practice of resource consolidation into SP systems creates a natural location for an Intranet Web server.

7.10.9 References

http://www.rs6000.ibm.com/resource/technology

http://www.specbench.org/osg/web96

http://w3.austin.ibm.com/cgi-bin/WebSz1.pl

7.11 Sizing Lotus Notes Servers

Sizing considerations for Lotus Notes Release 4 on AIX Version 4 are still difficult to give since information is in short supply. However, there are a few rules of thumb that might be used to size an AIX Lotus Notes Server.

Processor: The only way to determine processor size right now is to use NotesBench numbers. NotesBench is in its early stages at this time, but it is the only foundation we can use to get an idea of processor size.

First, the number of concurrent active users has to be determined. In existing Lotus Notes environments, this can be done by using the `show users` or `show tasks` commands from the Notes console. It is suggested to take the number of concurrent active users at five different times during peek periods and to determine the average of this number. If you do not have an existing Lotus Notes environment, you will have to take the total number of planned Lotus Notes users multiplied by an estimated ratio of concurrent users.

Second, you need to decide how many users belong to each NotesBench user class (Mail users, Mail & DB users, Groupware A users). See 6.9, “NotesBench Benchmark” on page 174.

Next, you will have to multiply the calculated workload by at least two in order to determine the sizing workload. This would mean that the customer's workload is about twice as heavy as the workload you will use for sizing. A good rule of thumb is that the actual workload will be two to four times that of the sizing workload. Now you will be able to determine the best processor from the published NotesBench data using the sizing workload.

Example:

Suppose your total number of Lotus Notes Users will be 2000. By using `show tasks`, you count the concurrent active users at five different times during a peek period: 700, 1200, 1700, 1000 and 900, for example. The average is 1100 concurrent active users. Now, you assume that 75 percent of this number are simple Mail users, 20 percent are Mail & DB users and 5 percent are Groupware A users. This means there are:

- 825 Mail users
- 220 Mail & DB users
- 55 Groupware A users

In order to determine the sizing workload, you will multiply those numbers by at least two, which leads to 1650 Mail users, 440 Mail & DB users, and 110 Groupware A users.

Now you can use the NotesBench data published by the hardware vendor and determine which numbers are closest to your needs. Most hardware vendors only publish NotesBench results for one or two user groups. Hardly any of them publish Groupware A results. In this case, you will have to work with the available numbers.

Allow room for growth, and factor in future upgrade paths, as Notes usage tends to explode once the users are comfortable with it.

You will also need to use the above-calculated number of users per group to determine the memory size.

Memory: Sizing memory for a Lotus Notes Server is probably the most critical point, since it will easily become a bottleneck when more Mail users start becoming active. Therefore, it is recommended to size memory rather generously. Keep in mind that AIX, with memory mapping, will take significant advantage of increased memory to improve response time.

Please be aware that the following explained procedures are only rules of thumb:

- Basic memory should be sized at 128 MB. This number considers AIX and the Lotus Notes Server itself.
- For every concurrent active Mail user (corresponds to Mail Test and Mail & Shared Database Test from NotesBench) doing simple transactions like sending regular-sized mail messages of about 1 KB, 0.25 MB of memory should be added.
- For heavy Notes users (corresponds to Groupware A from NotesBench) who perform transactions like sending large mail messages (of about 300 KB), attaching documents of 500 KB, doing replication from Notes Client to Notes Server, and executing `all document` searches (which means searches from all available documents in the database using a keyword), the sizing consideration is 1.3 MB for every concurrent active user.

This leads to the following formula:

Memory MB = 128 MB + (0.25 MB * CAUm) + (1.3 MB * CAUg)

CAUm = *Concurrent Active Mail & Database Users*
CAUg = *Concurrent Active Groupware A Users*

Of course, you will have to round up those numbers to the next configurable increment of memory.

For more-precise information about the workload of the different tests in the Lotus Notes Benchmark, please see 6.9, “NotesBench Benchmark” on page 174.

Disk: For disk sizing the recommendations can only be used as a rule of thumb:

Calculate approximately 100 MB for the required system files. This number will need to be increased as soon as Domino (Domino is the Web Browser for Lotus Notes) is integrated in Lotus Notes for AIX. Calculate an additional 500 MB for Lotus code libraries. For mail, 20-30 MB of disk space per user should be calculated. The rest of the database must be sized as appropriate for the individual situation. Here, as well, you need to keep in mind that the space needed for Lotus Notes will increase as soon as users start to get familiar with it.

Conclusion: All these sizing considerations are rules of thumb. They are meant to give some guidelines for sizing a Lotus Notes AIX Server.

Chapter 8. Performance Tools

Due to its UNIX heritage, AIX provides a powerful set of performance-monitoring and tuning tools. The available tools give performance information for different components of the system and on various parameters that affect performance. The details of the syntax and functions of these commands are documented in the *AIX V4.2 Commands Reference*.

Before using these tools a few concepts have to be clarified.

What is a Performance Bottleneck?: A performance bottleneck is the slowest component in a computer environment. This can either be a system resource like CPU, memory, or disk, or it could be the network. There is always a bottleneck because some resource will always be the slowest. The question is whether this bottleneck is a problem on a daily business.

How to Determine a Performance Bottleneck: The sequence of measuring system performance is extremely important. One should always follow the specified path, which is: CPU, Memory, I/O, Network.

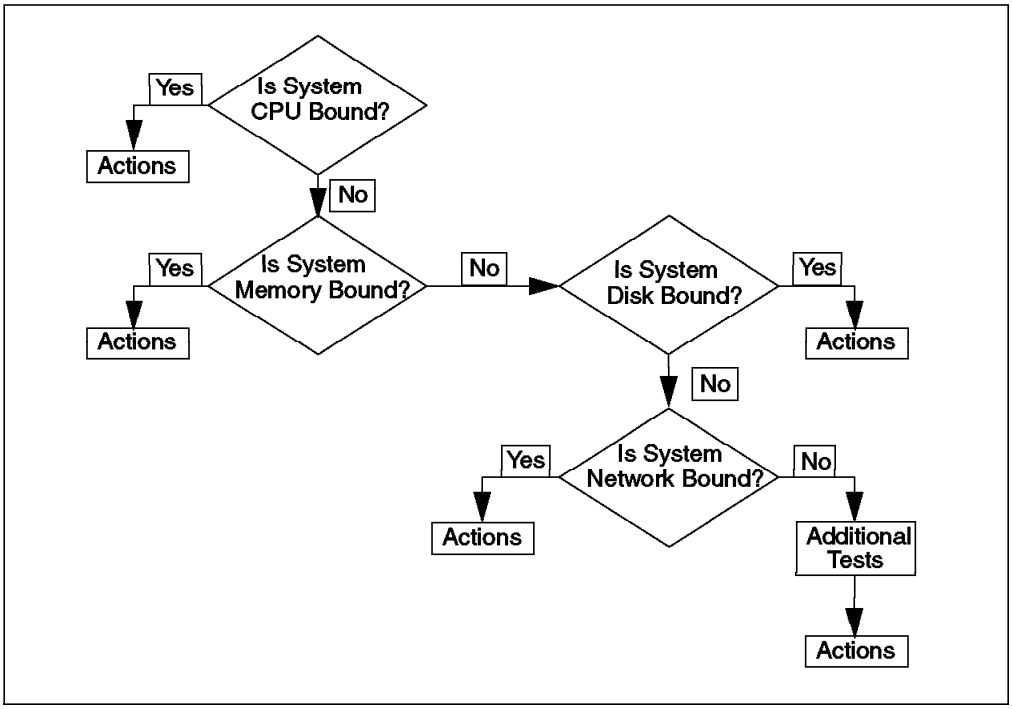


Figure 99. Performance Tuning Flowchart

Without following this path, you might overlook a bottleneck. For instance, if a system is paging a lot, you might see a heavy usage of the disks. Not looking at the memory statistics before looking at the I/O statistics could mislead you to the assumption that there is an I/O bottleneck.

Important

Always use the sequence as shown in the flowchart to determine a performance bottleneck:

- CPU
- Memory
- Disk I/O
- Network

8.1 AIX Performance Tools

As mentioned above, AIX provides several monitoring tools to determine a performance bottleneck. Some of them may also be used to tune the system. Not all of these tools come with the AIX Basic Operating System. Some of them are part of the AIX Performance Agent or AIX Performance Manager LPP software, as shown in Table 28.

	AIX V3.2	AIX V4
Base AIX	vmstat, iostat, sar, ps, netstat, nfsstat, trace, tprof, svmon, filemon, fileplace, netpmon, rmss, rmap, lvedit, schedtune, vmtune	vmstat, iostat, sar, ps, netstat, nfsstat, pdt, perfpmr, trace, schedtune, vmtune
Performance AIDE (Performance Agent)	xmservd	xmservd, tprof, svmon, filemon, fileplace, netpmon, rmss, bf, bfrpt, stem, syscalls, fdpr, lockstat
Performance Toolbox (Performance Manager)	xmperf, 3dmon, azizo, exmon, chmon	xmperf, 3dmon, azizo, exmon, chmon

Table 28. List of the Most Important Performance Tools

Often, specific performance tools can be used to monitor various system resources. Thus, it is hard to classify tools on the basis of a single system resource. Table 29 offers an overview of the performance tools.

	CPU	Memory	Disk	Network
Standard UNIX short range	vmstat, sar, ps	vmstat, ps	iostat, sar, vmstat	netstat, nfsstat
AIX-unique long range	Performance Toolbox	Performance Toolbox	Performance Toolbox	Performance Toolbox
AIX-unique short range	tprof	svmon, rmss	filemon, fileplace	netpmon
AIX-unique very detailed	stem, syscalls, fdpr, trace	bf, bfrpt, fdpr, trace	trace	trace

Table 29. List of Performance Tools by Resources

For detailed information on how to execute the commands and the meaning of the different metrics, please refer to the *AIX Command Reference*. Please

consider that the execution of each of the performance-monitoring tools will put some overhead on your system.

vmstat: The first tool to use is vmstat, which provides very quick and compact information about various system resources and their related performance problems. The vmstat command reports statistics about the number of kernel threads in the run queue and the wait queue, memory, paging, disks, interrupts, system calls, context switches, and CPU activity. The reported CPU activity is a percentage breakdown of user mode, system mode, idle time and waits for disk I/O. Keep in mind that the CPU statistics on SMP systems are an average of all processors. With vmstat**,** you cannot see a per-processor CPU usage on SMP systems.

The vmstat command adds little overhead to the system. It uses about 30 milliseconds of CPU time for each report generated.

When diagnosing a performance bottleneck, vmstat is only a first step. The system administrator will have to use more-complex monitoring tools to check the indications of vmstat and to determine whether there are possibly hidden performance bottlenecks.

iostat: The iostat tool reports CPU statistics and input/output statistics for TTY devices, disks, and CD-ROMs. It is used for monitoring system input/output device utilization by observing the time the physical disks are active in relation to their average transfer rates. The iostat command is useful to determine whether a physical volume is becoming a performance bottleneck and if there is a way to improve the situation.

The generated reports can be used to change system configurations to better-balance the input/output load between physical disks.

Since the CPU utilization statistics are also available with the iostat report, the percentage of time the CPU is in I/O wait can be determined at the same time. For multiprocessor systems, the CPU values are global averages among all processors. Also, the I/O wait state is defined system-wide and not per processor.

The iostat command adds little overhead to the system. It uses about 20 milliseconds of CPU time for each report generated.

Like vmstat, iostat cannot be used to finally decide whether there is a performance bottleneck. The system administrator will have to use more-complex tools like filemon to identify the source of the slowdown.

sar: The sar command reports either system-wide (global among all processors) CPU statistics (which are calculated as averages for values expressed as percentages, and as sums otherwise) or it reports statistics for each individual processor. Therefore, this command is particularly important on SMP systems. With its numerous options, sar also provides queuing, paging and TTY statistics. Only root is able to execute the sar command.

Note: The sar command only reports on local activities.

The sar command returns information such as the following:

- The number of kernel processes terminating per second.

- The number of times kernel processes could not be created because of enforcement of a process threshold limit.
- The number of kernel processes assigned to tasks per second.
- The number of IPC message primitives and semaphore primitives.
- Queue, paging, CPU and TTY statistics.

ps: The ps command displays statistics and status information about processes in the system, including process or thread ID, I/O activity, and CPU and memory utilization.

The ps command can be used to monitor memory use by an individual process. The ps v [pid] command provides reports on memory-related statistics for individual processes such as page faults. It also describes the size of a working segment that has been touched, the size of a working segment and code segment in memory, the size of a text segment, the size of a resident set, and the percentage of real memory used by this process.

The ps command provides standard output on the current status of active processes. If the -m flag is used, it also gives the status of associated kernel threads.

Note: You must use the -o THREAD flag in conjunction with the -m flag to display extra thread-related columns.

The CPU time consumed by this command varies with the number of processes to be displayed, but it usually does not exceed 0.3 seconds.

Only a snapshot is provided by ps. To gather data over time, use the tprof command.

netstat: Traditionally, netstat is used for determining network problems rather than for measuring performance. But it is useful in determining the amount of traffic on the network to ascertain whether performance problems are due to congestion.

The netstat command displays information regarding traffic on the configured network interfaces. Reports include:

- The address of any protocol control blocks associated with the sockets and the state of all sockets.
- The number of packets received, transmitted, and dropped in the communications subsystem.
- Cumulative statistics for errors, collisions, packets transferred.
- Routes and the status.

Most of the variations of this command use less than 0.2 seconds of CPU time.

nfsstat: The nfsstat command displays statistical information about NFS clients or servers and RPC calls. There is a specific parameter for the server information (nfsstat -s) and for the client information (nfsstat -c).

- NFS Server Information

  The NFS server displays the number of NFS calls received (*calls*) and rejected (*badcalls*), as well as the counts and percentages for the various kinds of calls made.

- NFS Client Information

  The NFS client displays the number of calls sent and rejected, as well as the number of times a client handle was received (*nclget*), the number of times a call had to sleep while awaiting a handle (*nclsleep*), and a count of the various kinds of calls and their respective percentages.

- RPC Statistics

  The `nfsstat` command displays statistical information pertaining to the ability of a client or server to receive calls. Information includes:

  – Total number of RPC calls received or rejected.
  – Number of times no RPC packet was available when trying to receive.
  – Number of packets that were too short or had malformed headers.
  – Total number of RPC calls sent or rejected by a server.
  – Number of times a call had to be transmitted again.
  – Number of times a reply did not match the call.
  – Number of times a call timed out.
  – Number of times a call had to wait on a busy client handle.
  – Number of times authentication information had to be refreshed.

tprof: The `tprof` tool is a very versatile AIX profiler that provides a detailed profile of CPU usage for every AIX process ID and name. It profiles at the application level, routine level and even the source statement level. This provides both a global view of the system as a whole and a detailed view of individual programs.

For subprogram level profiling, the `tprof` command can be run without modifying your executable program. You do not have to recompile with special compiler flags or linker options. This means that you can obtain a subprogram profile of any executable module that has already been built as long as it is not stripped. However, to get a profile at the source statement level, recompilation is required. Please note that `tprof` does not work with COBOL or PASCAL programs at the statement level, but it will work with C or FORTRAN programs.

The `tprof` command uses `trace`. Since only one trace (with the same trace hook) can run on an AIX system, you will not be able to use `trace`, `filemon` or `netpmon` simultaneously with `tprof`.

The `tprof` command will cause some system overhead, but since it only enables one trace hook, the degradation of the performance of a large compile, for example, would be less than 2 percent.

svmon: The `svmon` command offers a more in-depth analysis of memory usage. The `svmon` command displays information about the current state of memory. The displayed information does not constitute a true snapshot of the memory, because the `svmon` command runs at the user level with interrupts enabled. The `svmon` command creates four types of reports: global, process, segment, and detailed segment. The reports are:

- Global statistics describing the use of real memory.
- Statistics on the subset of real memory in use.
- Statistics on the subset of real memory containing pinned pages.
- Statistics describing the use of paging space.

The `svmon -G` command uses about 3.2 seconds of CPU time. An `svmon -P` command for a single process takes about 0.7 seconds of CPU time.

rmss: The reduced memory system simulator, `rmss`, offers a way to simulate RS/6000 systems with different sizes of real memory that are smaller than those of your actual machine (without having to extract and replace memory boards). Moreover, `rmss` provides a facility to run an application over a range of memory sizes, displaying, for each memory size, performance statistics such as the response time of the application and the amount of paging. In short, `rmss` helps answer either the question about how many megabytes of real memory are needed to run AIX and a given application or how many users can run this application simultaneously in a machine with X megabytes of real memory. It is important to keep in mind that the memory size simulated by `rmss` is the total size of the machine's real memory, including the memory used by AIX and any other programs that may be running. It is not the amount of memory used specifically by the application itself. Because of the performance degradation it can cause, `rmss` can be used only by root or a member of the system group.

filemon: The `filemon` command collects and presents trace data on the various layers of t he file system, and it reports the I/O activity on behalf of logical files, virtual memory segments, logical volumes, and physical volumes. It will most likely be used if you assume that you have an I/O bottleneck on your system.

To provide a more-complete understanding of file system performance for an application, the `filemon` command monitors file and I/O activity at four levels:

Logical file system	The `filemon` command monitors logical I/O operations on logical files. The monitored operations include all *read*, *write*, *open*, and *lseek* system calls, which may or may not result in actual physical I/O, depending on whether or not the files are already buffered in memory. I/O statistics are kept on a per-file basis.
Virtual memory system	The `filemon` command monitors physical I/O operations (such as paging) between segments and their images on disk. I/O statistics are kept on a per-segment basis.
Logical volume	The `filemon` command monitors I/O operations on logical volumes. I/O statistics are kept on a per-logical-volume basis.
Physical volumes	The `filemon` command monitors I/O operations on physical volumes. At this level, physical resource utilizations are obtained. I/O statistics are kept on a per-physical-volume basis.

In its normal mode, the `filemon` command runs in the background while one or more application programs or system commands are being executed and monitored. The filemon command automatically starts and monitors a trace of the program's file system and I/O events in real time.

Like `tprof`, the `filemon` command uses the AIX system trace facility. Currently, the trace facility only supports one output stream. Consequently, only one `filemon` or `trace` process can be active at a time. If another `filemon` or `trace` process (like `tprof`) is already running, the `filemon` command will respond with an error message.

The performance impact of `filemon` depends on how low or high the I/O rate on the system is. The CPU consumption can range from 1 percent at a low I/O rate to 5 percent at a high I/O rate.

fileplace: The `fileplace` command displays the placement of blocks for a specified file within the AIX logical or physical volumes containing the file. It will most likely be used if the `filemon` command displays large seek distances.

Optionally, the `fileplace` command will also display:

- Statistics indicating the degree to which the file is spread within the volume.
- The indirect block addresses for the file.
- The file's placement on physical (as opposed to logical) volume, for each of the physical copies of the file.

Most variations of this command use less than 0.3 seconds of CPU time.

netpmon: The `netpmon` command is used to determine network-related performance problems. It monitors a trace of system events, reporting on network activity, performance, and network-related CPU usage during the monitored interval. By default, `netpmon` runs in the background while one or more application programs or system commands is being executed and monitored.

The `netpmon` command reports in detail on the following system activities:

CPU usage	The `netpmon` command monitors CPU usage by all threads and interrupt handlers. It estimates how much of this usage is due to network-related activities.
Network device driver I/O	The `netpmon` command monitors I/O operations through all Ethernet, token ring, and fiber distributed data interface (FDDI) network device drivers. In the case of transmission I/O, the command also monitors utilizations, queue lengths, and destination hosts. For received I/O, the command also monitors time in the demux layer.

With a moderate, network-oriented workload, `netpmon` increases the overall CPU utilization by 3-5 percent.

fdpr: The `fdpr` command (feedback directed program restructuring) is a performance-tuning utility that may help improve the execution time and the real memory utilization of user-level application programs. The `fdpr` command optimizes the executable image of a program by collecting information on the behavior of the program while it is used for some typical workload. The `fdpr` command then creates a new version of the program that is optimized for that workload. The new program generated by `fdpr` will typically run faster and use less real memory.

Important

The new executable will not be supported by IBM.

You should run a full test cycle for the new executable to guarantee the same functionality.

Performance will change when the initial workload, data or parameters change.

bf: The bf (bigfoot) command traces the memory use of the applications. With the back-end program bfrpt, it provides a detailed trace, or footprint, of the memory page references for most processes on the system. It captures references to all unpinned pages and many pinned pages.

stem: The stem (scanning tunneling encapsulating microscope) command is a tool for inserting instrumentation code (subroutines), either user-supplied or default routines provided with stem. The stem command operates on existing libraries and programs without requiring source code or recompilation of the libraries or programs as long as the program is not stripped.

syscalls: The syscalls (system call tracing) command captures system call entry and exit events by individual processes or for all processes on the system. The syscalls command also can maintain counts for all system calls over long periods of time.

trace: The AIX trace facility is useful for observing system activity, particularly that of a running device driver. The trace facility captures a sequential flow of time-stamped system events, providing a fine level of detail on system activity. Events are shown in time sequence and in the context of other events. The trace facility is useful in expanding the trace event information to understand which, when, how, and even why the event happened. The operating system is shipped with permanent trace event points. These events provide general visibility to system execution. You can extend the visibility into applications by inserting additional events and providing formatting rules with low overhead. Because of this, the facility is useful as a performance-analysis tool and as a problem-determination tool. The trace facility is more flexible than traditional system-monitor services that access and present statistics maintained by the system. With traditional monitor services, data reduction (conversion of system events to statistics) is largely coupled to the system instrumentation. For example, the system can maintain the minimum, maximum, and average elapsed time observed for runs of a task and permit this information to be extracted.

The trace facility does not strongly couple data reduction to instrumentation, but it does provide a stream of system events. It is not required to presuppose what statistics are needed. The statistics or data reduction are to a large degree separated from the instrumentation. You can choose to develop the minimum, maximum, and average time for task A from the flow of events. But it is also possible to extract the average time for task A when called by process B, extract the average time for task A when conditions XYZ are met, or even decide that some other task, recognized by a stream of events, is more meaningful to summarize. This flexibility is important for diagnosing performance or functional problems. For example: netpmon uses the trace to report on network activity, including CPU consumption, data rates and response time. The tprof command uses the trace to report the CPU consumption of kernel services, library subroutines, application program modules, and individual lines of source code in the application program.

PDT: The performance diagnostic tool (PDT) collects configuration and performance information. It attempts to identify potential problems, both current and future. In assessing the configuration and the historical record of performance measurements, PDT attempts to identify:

- Unbalanced use of resources or asymmetrical aspects of configuration or device utilization. In general, if there are several resources of the same type, then performance is improved by meeting the following goals for a balanced use of those resources:

  - Comparable numbers of physical volumes (disks) on each disk adapter.
  - Paging space distributed across multiple physical volumes.
  - Roughly equal measured load on different physical volumes.
- Trends in usage levels that will lead to saturation. Resources have limits to their use. Trends that would attempt to exceed those limits should be detected and reported. A disk drive cannot be utilized more than 100 percent of the time. File and file system sizes cannot exceed the allocated space.
- New consumers of resource-expensive processes that have not been observed before. Trends can indicate a change in the nature of the workload as well as increases in the amount of resource used:
  - Number of users logged on.
  - Total number of processes.
  - CPU idle percentage.
- Inappropriate system parameter value settings that may cause problems.
- Hardware or software errors that may lead to performance problems. So it checks the hardware and software error logs and reports bad VMM pages.

PerfPMR: The PerfPMR package was developed to ensure that reports of suspected performance problems in AIX were accompanied by enough data to permit problem diagnosis by IBM. This makes the shell scripts that the command `perfpmr` invokes useful to other performance analysts, as well.

The `perfpmr` command collects data configuration and load averages followed by the execution of `monitor`, which runs `sar, iostat, vmstat, netstat` and `nfsstat` periodically. An instance of `trace` collects system trace data about everything happening on the system while `tprof` collects CPU usage data. The `filemon` command monitors the file system usage, and `netpmon` collects data about network activity.

The `perfpmr` command generates files containing statistics for each interval (*.int*) and summary (*.sum*) data from the interval files.

Other tools: Other commands are available for tuning your system resources. The system administrator can, for example, change the values of virtual memory manager (VMM) memory load control parameters, the CPU timeslice duration, and the paging space low retry interval with the `schedtune` command. Moreover, the system administrator can change the VMM page replacement algorithm parameters with the `vmtune` command, change the values of network options with the `no` command, change the priority of running processes with `renice` or change the priority of programs being started with `nice`.

8.1.1 SMP Tools

There are a variety of tools, either included with AIX V4 or available separately, that can help you in monitoring your SMP system. The objective of this section is to introduce the major performance tools that are available.

SMP tools available in AIX V4 can be categorized as follows:

- Standard tools or commands that were available in AIX V3.2 and have been modified to support the SMP environment.
- Tools that are SMP-specific and new in AIX V4.

The following tools were modified from AIX V3.2 to allow gathering of information about threads in an AIX V4 system. As much as possible, the standard options of each tool have been manipulated in order to support SMP and threads. New options have only been created where the existing tool options did not fulfill the new needs.

`ps` This command gives processes and threads status.

`pstat` This command displays related threads information.

`sar` This command shows the system activity and CPU activity for all the processors or for a specific processor.

`vmstat` This command displays system activity (memory and CPU usage) for all the processors.

`iostat` This command shows the balance between the disks.

The following tools are SMP specific:

`cpu_state` Command that shows the number of processors and the current state of each processor within the system. A processor may be enabled, disabled or unknown. This command is part of the bos.rte.mp fileset (bos.mp in AIX V4.2).

`bindprocessor` Is used to distribute the workload on the system. It enables the binding of individual processes to a processor within the SMP system. This command is also part of the bos.mp fileset.

`lockstat` Command that provides information on kernel locks caused by the current contention between threads on the system.

Here is a short description of the SMP tools that either were not described in 8.1, "AIX Performance Tools" on page 260 or have a major change important for SMP:

pstat: The `pstat` command is the nonintegrated form of the `crash` command. Because it has a number of new options, `pstat` is useful for looking at threads.

`-A` Shows all entries in the kernel thread table

`-P` Shows runnable kernel threads only

`-U <proc number>` Shows thread slot user structure of kernel threads for any given processor

`-S` Shows processor status (which thread is running on which processor at the time of the command)

sar: The processors' load can be measured with the `sar` command. The syntax of the `sar` command for this task is as follows:

```
sar [ -P <processor_id> [, ... ] | ALL ] <interval> <count>
```

Use of `sar -A -P ALL <x> <y>`, where `<x>`=time interval in seconds and `<y>`=number of iterations, shows information about all the SMP-related counters, such as forks, characters read, and characters written per processor.

Using `-P <processor_id>` reports per-processor statistics for the specified processor or processors separated by a comma.

The `ALL` keyword reports statistics for each individual processor and globally for all processors. Of the flags that specify the statistics to be reported, only the `-a, -c, -m,-u,` and `-w` flags are meaningful with the `-P` flag. Meaningless flags are silently ignored.

cpu_state: Enabling or disabling processors can also be used to size a system for a specific application and to determine the number of processors that are needed to run the application with good performance. In either case, the `cpu_state` command can be used to measure the scalability of a system or size a system.

The `cpu_state` command lists and controls which processors on a multiprocessor system will be active when the system is next started. It is the only command that shows all physically present processors.

bindprocessor: AIX V4 allows users to bind processes to a specific processor by using the `bindprocessor` command. Although the `bindprocessor` command is intended for tuning, it is very useful for performance analysis on SMP systems. The bound process will run only on the designated processor. If the process is multithreaded, all the related threads will be bound to the same processor.

Binding is different from partitioning; it is not possible, for example, to dedicate a set of processors to a specific workload and another set of processors to another workload.

It is not possible to bind a thread to a specific processor. You can bind one or several threads within a process at the programming level by using the `bindprocessor(What,Who,Processor)` call. The `bindprocessor()` call must be used in the source code of your program.

lockstat: In AIX, you can use the `lockstat` command to see the use of locks. Only kernel locks can be seen with the `lockstat` command.

The `lockstat` command supports the use of user-supplied lock names in files named `/usr/include/sys/lockname_*.h`, where * is a wildcard.

If `vmstat` indicates that there is a significant amount of CPU idle time when the system seems subjectively to be running slowly, delays may be due to kernel lock contention.

In AIX V4, this possibility should be investigated with the `lockstat` command.

Look for the following pointers:

- Check `lockstat` output for Ref/s > 10000
- Identify subsystems and lock classes that have a high number of Ref/s

Application problems can only be seen indirectly. If there is lock contention, you must check for bottlenecks caused by the application.

For example, if your application has a high number of processes that read and write in a unique message queue, you might have lock contention for the virtual memory manager (VMM) subsystem. Adding more message queues may reduce the level of lock contention.

8.1.2 Additional Trace-Based Tools

The following trace-based tools were originally used in the development of AIX but have now been made generally available. They can be obtained on the Internet, via anonymous FTP, from:

`ftp.software.ibm.com`

in the directory: `/aix/tools/perftools`

`utld` Reports on system locks and delays. Although intended for use on all AIX platforms, this tool is especially useful in an SMP environment. It reports thread/processor affinity and gives a greater insight into locking than can be gained by using tools such as `lockstat`. This tool also provides a very useful summary of system utilization.

`par` Reports on disk I/O transfers and utilization. The report produced contains three sections:

- Summary of all files referenced per process
- Logical read/write times per process/file
- Physical transfer times and utilization per disk

`why` Reports the extent and reasons why processes/threads are delayed in their execution. It can also provide an optional report to examine the correlation between execution of a given thread and execution of all other threads in the system. In particular, this option may be used to examine the state of all processes when the idle process(es) executes.

Further documentation on all these commands is included in their respective packages.

Important

Please note that these tools are currently available free of charge and, as such, are provided without warranty or support. As they are development tools, they may change significantly in future releases.

8.1.3 References

AIX Versions 3.2 and 4 Performance Tuning Guide, SC23-2365

AIX Command Reference, SBOF-1851

8.2 Performance Toolbox for AIX (PTX/6000)

Anyone faced with the task of keeping a network of computers well tuned and capable of performing as expected recognizes the need for a comprehensive tool for monitoring and tuning system performance.

The Performance Toolbox (PTX) for AIX is a set of tools to monitor the performance of UNIX platforms. PTX also aids in the tuning of such UNIX platforms and offers a wide range of performance tools within a versatile framework for both stand-alone and networked systems.

PTX is designed to properly answer questions about performance such as: “How can I tell my system is performing optimally?”, “Does the system meet my performance expectations?”, “Why is the system response time so slow lately?”, “Why do my applications take so much longer to run?” or “Which disk drive is a bottleneck in the I/O workload?”.

8.2.1 Performance Toolbox Concepts

PTX for AIX is a client/server tool set based on Motif. It monitors local and remote system performance with several graphical windows that are fully user configurable. This includes 2D and 3D views of performance statistics.

PTX for AIX is divided in two components: agent and manager.

The manager has the tools for the key elements of performance monitoring and tuning:

- Monitoring of system statistics.
- Analysis of system statistics.
- Tuning of the system performance parameters to balance the utilization of fixed system resources.

The agent's part consists of programs that run on the machines you want to monitor. The role of the programs is to get and filter data that will be handled by the manager.

The client/server implementation lets PTX be an excellent aid to monitor and tune the performance of various UNIX systems in a network environment from a single graphics workstation.

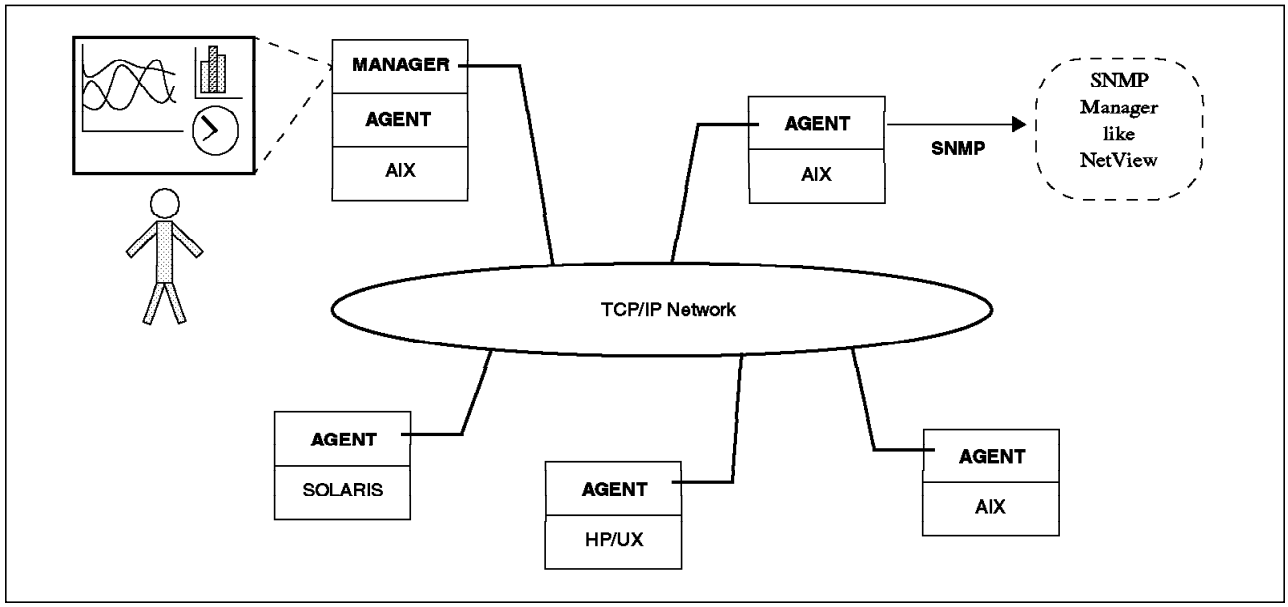


Figure 100. Performance Toolbox Environment

While you may use a graphics display connected to the server, we strongly recommend that you use a separate graphics workstation. This will minimize the impact of running PTX on the system load, and it will give a truer indication of system performance.

Major PTX features are:

- Concurrent monitoring of near-real-time local and remote machine statistics.
- Configurable color graphic display format in user-designed monitoring consoles.
- Extensive record/playback/analysis for long-term performance analysis.
- Data filter/alarm facilities and exception monitoring.

- Separately installable manager and agent components that allow a manager to monitor multiple agents and an agent to supply data to multiple managers.
- HP/UX (V9.03), SunOS (V4.1.3) and Solaris (V2.4) agents to allow monitoring of performance data on OEM machines.
- Application programming interfaces (APIs) that allow programmers to access local or remote data as well as to register custom data with the local agent.
- Ability to respond to SNMP "get" and "get next" requests and to send traps to an SNMP manager (AIX agents only).
- Support for RS/6000 SP systems using the Performance Toolbox Parallel Extensions Feature of the Parallel System Support Programs (PSSP) for AIX/6000.

Performance Toolbox Parallel Extensions (PTPE) is a feature of PSSP 2.2 and is used in combination with PTX to simplify performance analysis and reduce PTX administrative overhead in RS/6000 SP systems by organizing the SP nodes into reporting groups. It provides the utilities to monitor, store and retrieve performance information collected on RS/6000 SP subsystems such as the high performance switch, virtual shared disk and LoadLeveler.

8.2.2 Graphical Monitoring and Analysis Issues

System Monitoring: In network client/server applications, the performance of sets of systems working together can be as important as the performance of an individual system. Likewise, the performance of multiple applications working together can be as important as that of an individual application. Therefore, it is very important to be able to get the "big picture" by graphically viewing many correlated parameters concurrently across multiple nodes in a network. PTX/6000 allows a user to concurrently visualize the live (near-real-time) performance characteristics of the clients and server applications across the network.

Analysis and Control: By providing an umbrella for tools that can be used to analyze performance data and control system resources, the manager program `xmperf` assists the system administrator in keeping track of available tools and in applying them in appropriate ways. This is done through a customizable menu interface. Tools can be added to menus, either with fixed sets of command-line arguments to match specific situations or in a dialog window. The menus of `xmperf` are preconfigured to include most of the performance tools shipped as part of the tools option of the agent component.

All performance-related tools already available in AIX can be accessed through this interface. In addition, the ability to record load scenarios and play them back in graphical windows at any desired speed offers new ways of analyzing performance problems.

Features for analyzing a recording of performance data are provided by the `azizo` program and its support programs. Recordings can be produced from the monitoring programs `xmperf` and `3dmon` during monitoring, or they can be created by the `xmservd` daemon. The `xmservd` daemon allows for recording with a minimum of overhead. This makes constant recording possible so that you can analyze performance problems after they have occurred.

Finally, using the agent component filter `filtd`, you can define conditions that, when met, could trigger any action you deem appropriate, including alerting yourself and/or

initiating corrective action without human intervention. This facility is entirely configurable so that alarms and actions can be customized to your installation.

Capacity Planning: If you can make your system simulate a future load scenario, `xmperf` can be used to visualize the resulting performance of your system. By simulating the load scenario on systems with more resources, such as more memory or more disks, the result of increasing the resources can be demonstrated.

Network Operation: The `xmservd` data-supplier daemon can provide consumers of performance statistics with a stream of data. Frequency and contents of each packet of performance data are determined by the consumer program. Any consumer program can access performance data from the local host and one or more remote hosts. Any data-supplier daemon can supply data to multiple hosts.

SNMP Interface: By entering a single keyword in a configuration file, the data-supplier daemon can be told to export all its statistics to a local `snmpd` SNMP agent. Users of an SNMP manager such as IBM NetView see the exported statistical data as an extension of the set of data already available from `snmpd`.

Note: The SNMP multiplex interface is only available on IBM RS/6000 agents.

8.2.3 Manager

The manager component of the Performance Toolbox for AIX has the following components:

`xmperf` The main interface program providing graphical display of local and remote performance information in a menu interface to commands of your choice.

`3dmon` A program that can monitor up to 576 statistics simultaneously and display the statistics in a three-dimensional graph.

`3dplay` A program to play back `3dmon` recordings in a `3dmon`-like view.

`chmon` Supplied as an executable as well as in source form. This program allows monitoring of vital statistics from a character terminal.

`exmon` A program that allows monitoring of alarms generated by the `filtd` daemon running on local or remote hosts.

`azizo` A program that allows you to analyze any recording of performance data. It lets you zoom in on sections of the recording and provides graphical as well as tabular views of the entire recording or zoomed-in parts.

`ptxtab` A program that can format recording files for printed output.

`ptxmerge` This program allows you to merge up to 10 recording files into one. For example, you could merge `xmservd` recordings from the client and server sides of an application into one file to better correlate the performance impact of the application on the two sides.

The xmperf Program: The `xmperf` program is the most comprehensive and largest program in the manager component. It is based on the X Window system and was developed with the OSF/Motif toolkit. The `xmperf` program allows you to define monitoring environments to supervise the performance of the local AIX system and remote AIX, SUN or HP/UX systems.

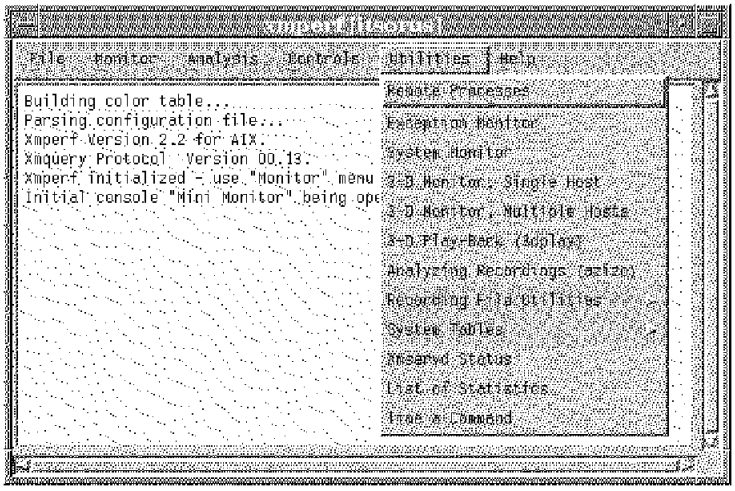


Figure 101. AIX Performance Toolbox Initial Screen

Each monitoring environment consists of a number of consoles displayed as graphical windows. Consoles, in turn, contain one or more instruments, and each instrument can show one or more values that are monitored (see Figure 103 on page 276).

The following terms are used to refer to the `xmperf` monitoring functions or components:

- A *console* is a graphical window containing instruments that monitor the system. A console can have one or more instruments.
- An *instrument* is a graphical view of monitored values, and each instrument can show one or more values that are monitored. The presentation of the values can be in form of graphs, gauges and so on.
- A *value* is the unit to be monitored. It can be any quantifiable aspect of system performance, including CPU usage for user processes, disk transfer rates or TCP bytes transmitted.
- *Groups of statistics* are a functional part of the system. The values are grouped in relation to the functional part of the system they belong to. For example, CPU global user, kernel, wait and idle percentages are a group of CPU statistics. However, an instrument can have values from several groups. The groups of statistics are shown in Figure 102 on page 275.

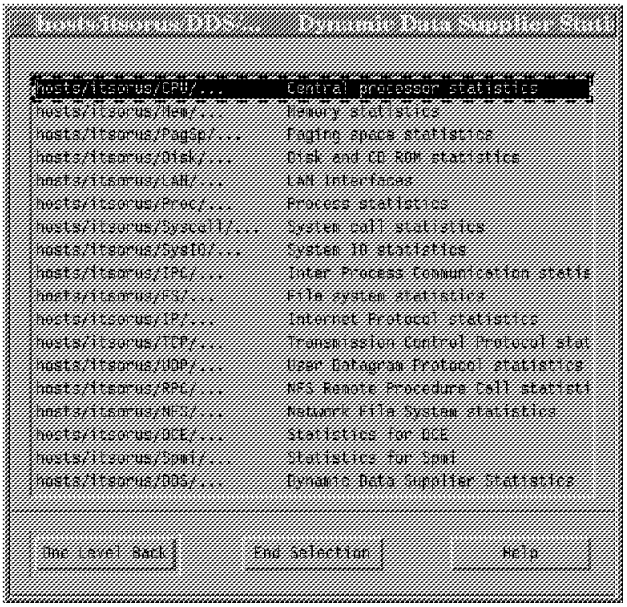


Figure 102. Groups of Statistics

The `xmperf` command is not hard-coded to monitor a fixed set of resources. It is dynamic in the sense that a system administrator can customize it to focus on exactly the resources that are critical for each host that must be monitored.

Consoles can be told to do a recording of the data they monitor to disk. Such recordings can be played back with `xmperf` and analyzed with the `azizo` program.

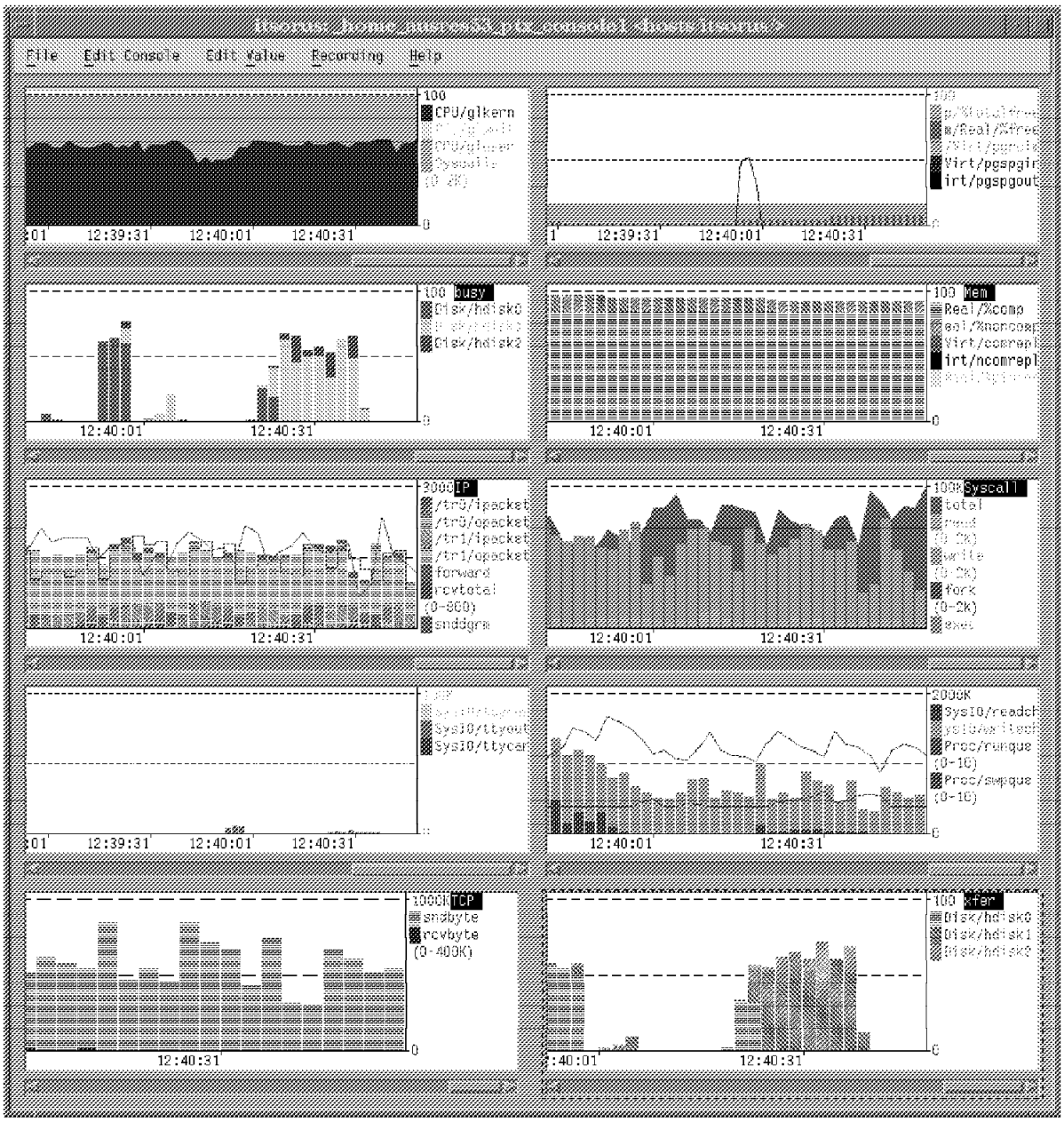


Figure 103. A User's Console

The 3dmon and 3dplay Utilities: The 3dmon program offers a graphical presentation that allows you to survey the same key data on several hosts. The output looks like a chessboard, except that each side may have from one to 24 fields.

If you use 3dmon so that it will display data from several hosts, it will emit a request at launch time for any available agents on the network and then lets you choose those you want to survey. The 3dmon program is really helpful for monitoring all the active CPUs in the network.

It can be configured to contact specific hosts -- up to 24 at one time -- and to display 24 values for a maximum of 576 simultaneous statistics.

With 3dmon, it is possible to record any 3dmon console for future analysis using the 3dplay program; 3dmon allows the user to add notes to a record. With 3dplay you can use seek and speed controls like those found on a conventional VCR.

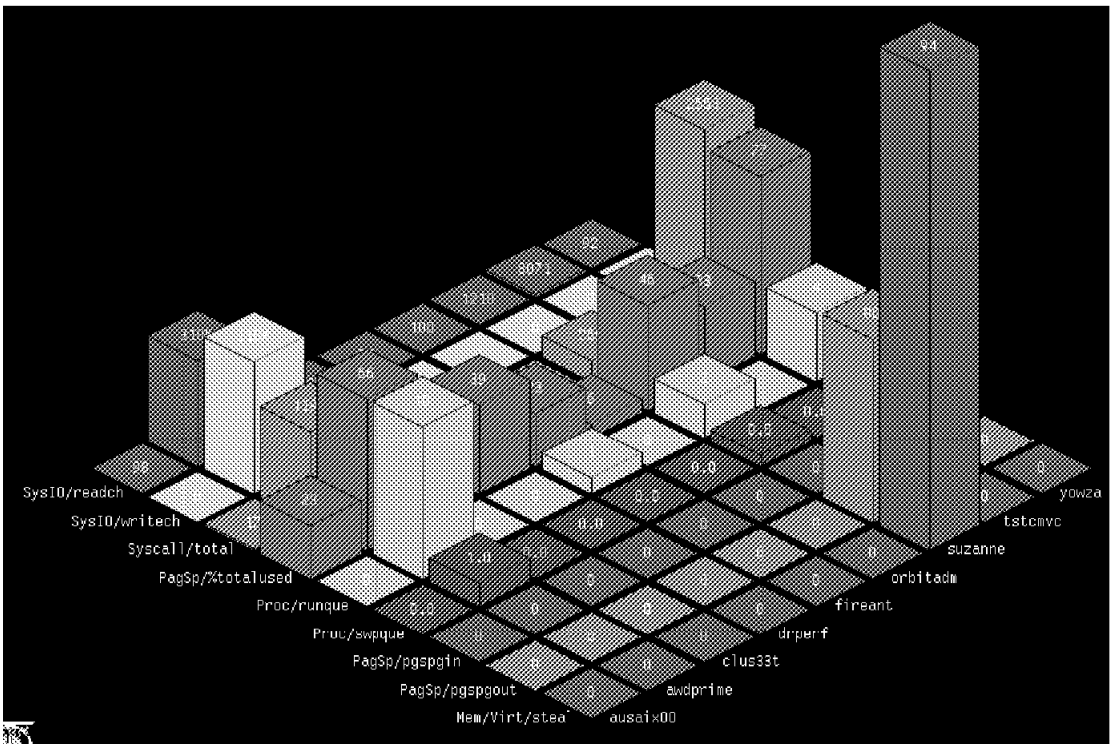


Figure 104. A View of 3dmon

The azizo and ptxmerge Utilities: The azizo tool is a PTX program that is used to analyze prerecorded files. It can analyze only one file at a time and render a maximum of 256 statistics (see Figure 105 on page 278). If multiple recordings must be analyzed together, the support program ptxmerge can be used to merge multiple recording files into one for simultaneous analysis of statistics from multiple sources.

The azizo tool provides statistics on the file, number of samples, time stamp, and minimum, maximum, average, and standard deviation for each value recorded. With azizo, it is possible to build subgraphs from the original graph by removing values or changing start and/or end times. These subgraphs can be saved for further analysis.

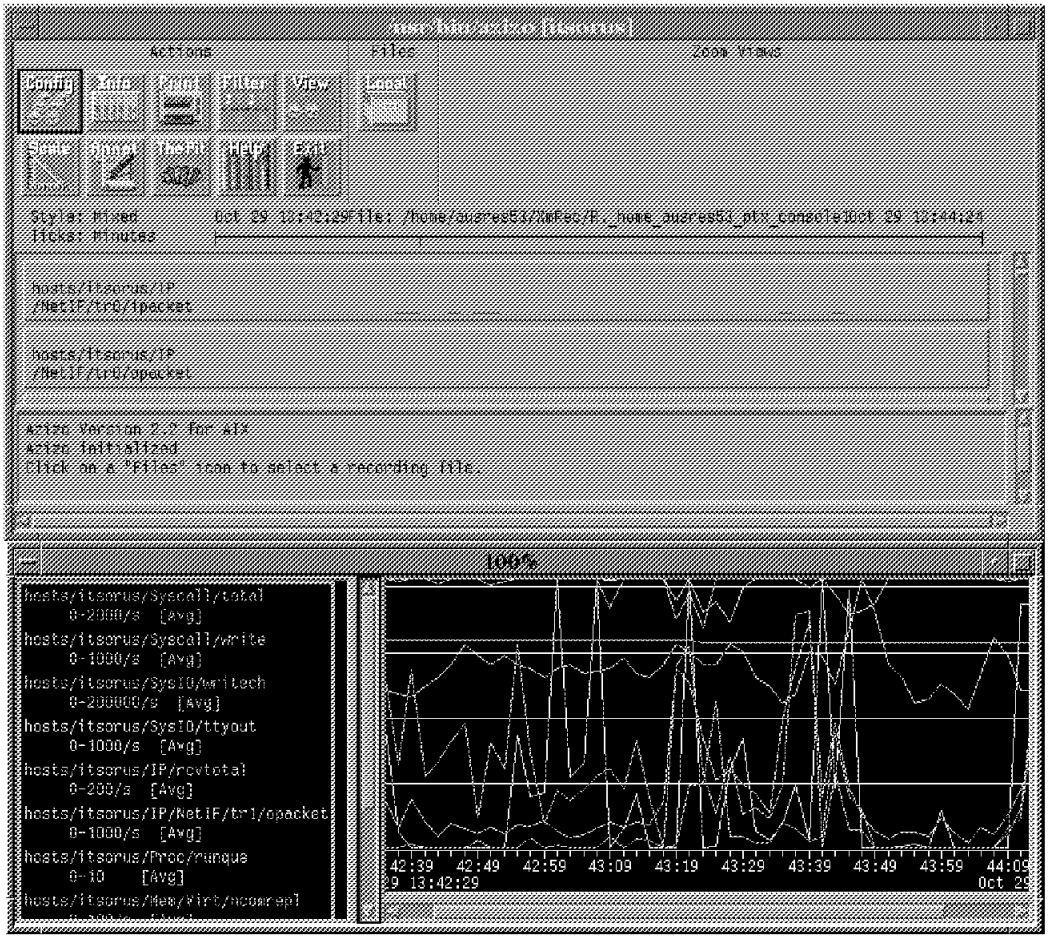


Figure 105. Azizo Interface

The exmon Utility: The exception monitor program, exmon, is a tool to manage exceptions. Exceptions are numbered from 0 to 10 -- they are packets transmitted from the agents on the networks for consumers who request them. The exmon program is one of these consumers.

The exmon program is designed to provide a convenient facility for monitoring exceptions as they are detected on remote hosts. It does so by allowing its user to register subscriptions for exceptions packets from all or selected hosts in the network and to monitor the exception status in a graphical window.

The chmon Program: The chmon program allows you to display data on a text-based screen for a given host. Default data are CPU, memory, disks, and system calls activities. It also provides the data consumed by the most active processes on the machine.

The chmon program is a sample of using one of the two APIs provided with PTX, the *Rsi* API. It can be modified to display requested values in any useful format. A sample of a chmon screen is shown in the Figure 106 on page 279.

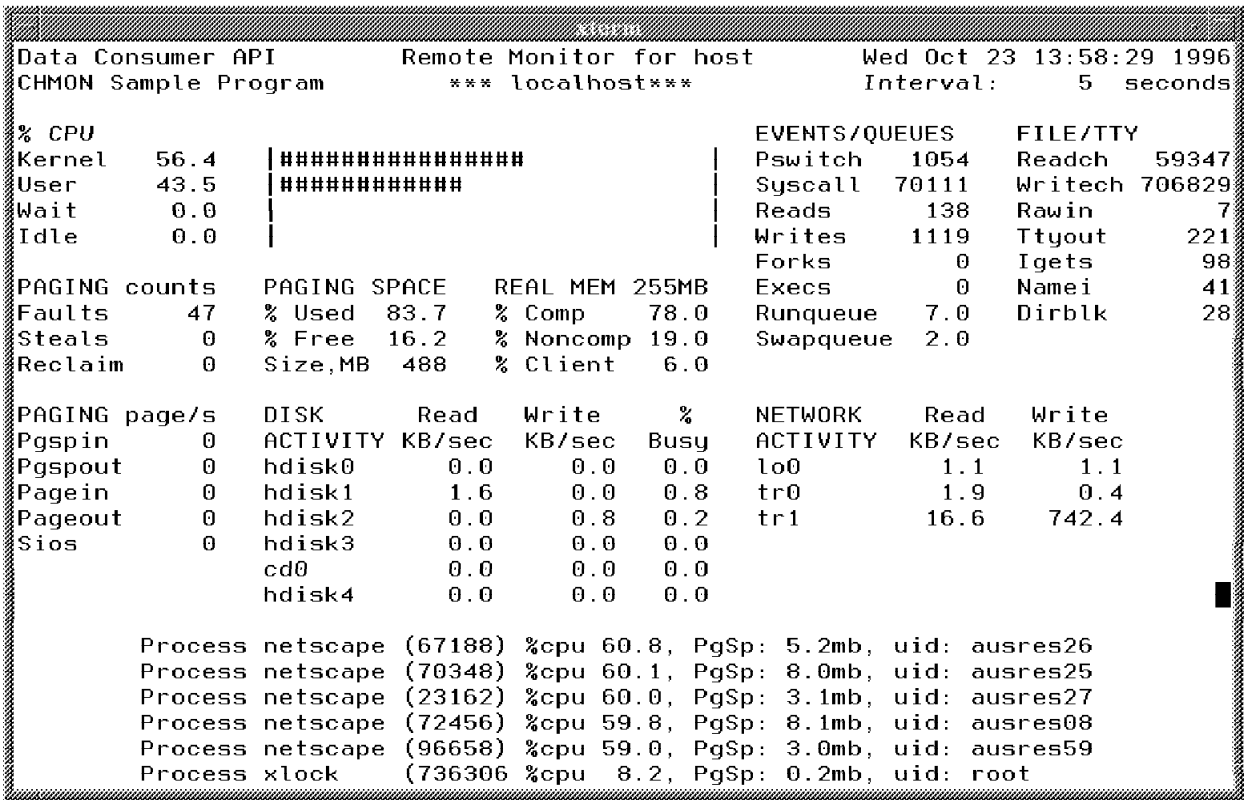


Figure 106. Chmon Illustration

8.2.4 Agent

The agent component is a collection of programs that make it possible for a host to act as a provider of performance statistics across a network or locally. The key program is the daemon `xmservd`. It supplies the statistics for the monitoring environment through an API called system performance measurement interface (SPMI). The SPMI implementation allows one agent to supply data to many managers, and it allows one manager to request data from many agents. The SPMI interface can be used for any dynamic data-supplier program to export its data.

Another agent program is `filtd`. This daemon is in charge of filtering data by processing all previously defined expressions that define new statistics. This feature allows you to easily combine existing “raw” statistics into new statistics that make more sense in the monitoring environment. The `filtd` also supplies `exmon` with the exceptions that occurred in the monitored system, and it allows you to define alarms that trigger actions.

The `xmservd` daemon also acts as a supplier of performance statistics to simple network management protocol (SNMP) managers like NetView. This feature is only available in RS/6000 systems.

8.2.5 Monitoring an SMP with the Performance Toolbox

PTX can be used to monitor an SMP system.

In order to monitor an SMP with Performance Toolbox, you need a graphical display. While you may use a graphical display connected to some SMPs, we again strongly recommend that you use a separate graphics workstation. This will minimize the impact of running PTX on the system load, and it will give a truer indication of system performance.

PTX provides predefined consoles for monitoring a single system. All you need is to create your own console using the tools provided in the main window of `xmperf`.

You can select a value and then customize the properties of the values you have selected, such as the color and the type of graph you want (line, area, bars). You will be able to set upper and lower limits, set a threshold, and set an alarm when this threshold is reached.

Select values you want to monitor for processor 0. You can then edit your console and add a new local instrument for processor 1, and so on. Figure 107 on page 281 shows an example of a customized SMP console.

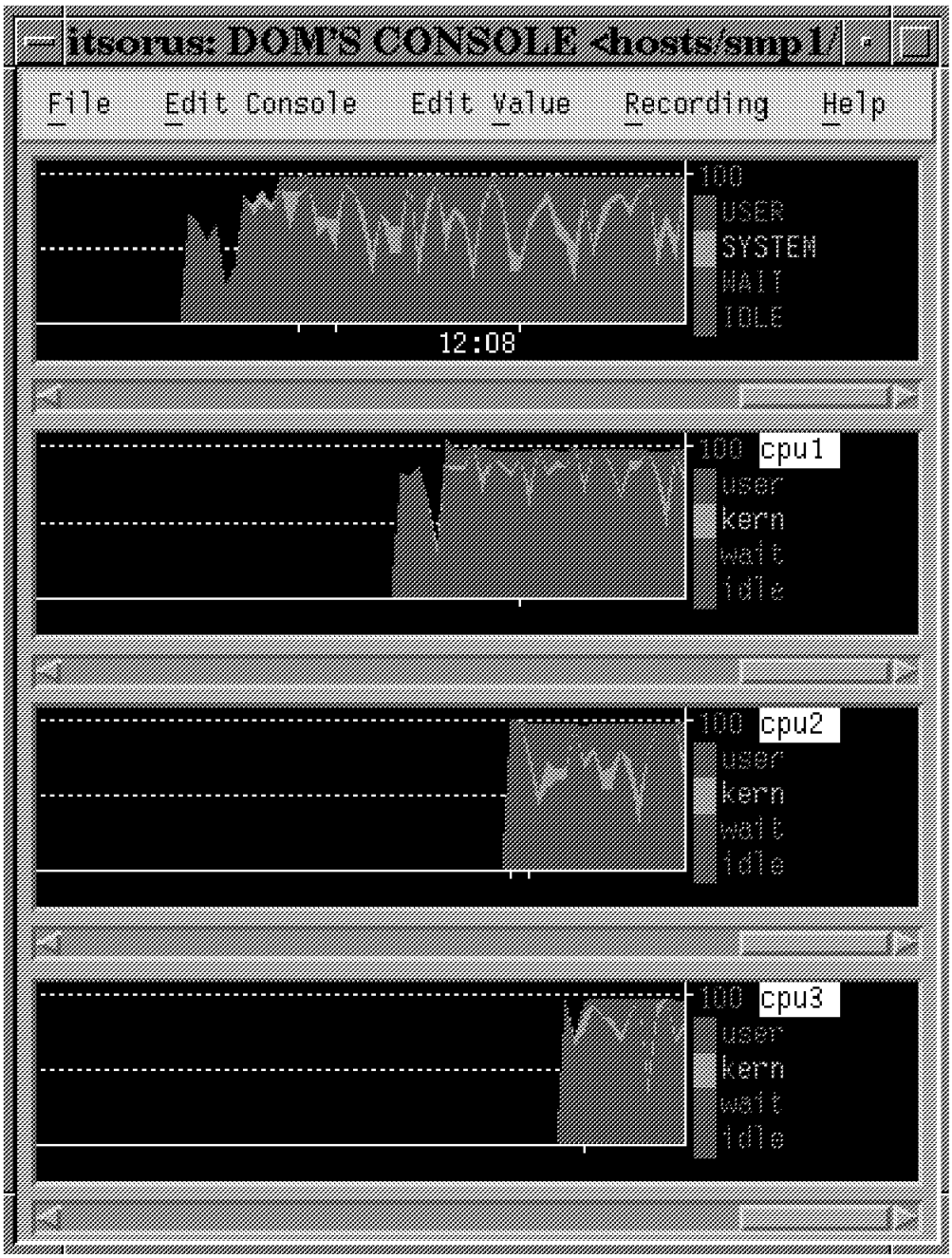


Figure 107. SMP Console Example

Monitoring an SMP with 3dmon: The 3dmon program provides a quick method of producing the same results in a three-dimensional view for important performance values.

This monitor may be invoked by going to the utilities menu in the main xmperf window. Inside this menu, you will find both the 3-D Monitor, Single Host and 3-D Monitor, Multiple Hosts submenus. Choosing Local Processors (CPUs) will give you a screen where you can choose which CPUs you want to monitor, and when complete, your screen is displayed. The performance values you will be monitoring are: user, kern, wait, pswitch, syscall, read, write, fork, exec, readch, writech, iget, namei, and *dirblk*.

The 3dmon monitor may also be invoked from the command line by typing:

```
# 3dmon -h <hostname>
```

At this step, you can select resources you want to monitor and change the sampling interval. If you select Local Processors (CPUs), you will then see the following screen:

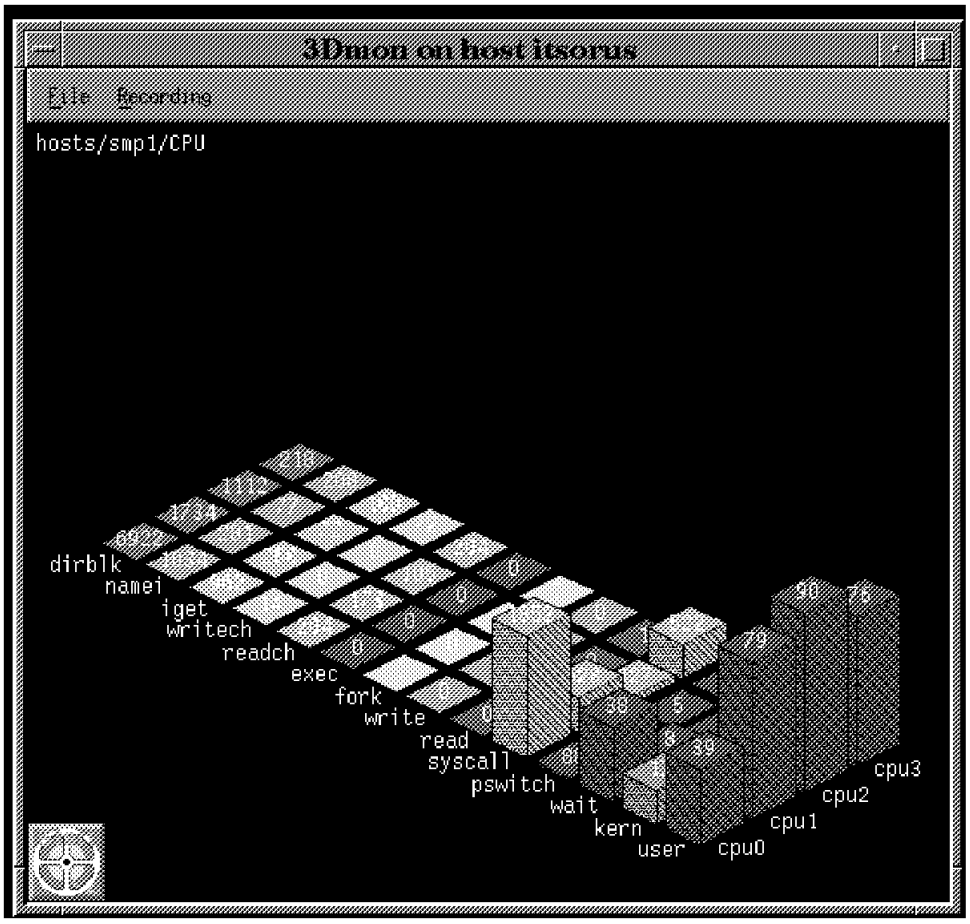


Figure 108. 3dmon Output on a 4-Way SMP

Note: If you cannot read the values behind the first towers corresponding to the *user* activity, you can move any monitored value to the front by double-clicking on the name of that value. For example, if you want to read the *kern* values for all the processors, you can double-click on *kern*. It will then move to the first position.

8.2.6 References

Performance Toolbox for AIX Guide and Reference, Version 1.2 and 2, SC23-2625

http://www.developer.ibm.com/library/aixpert/nov93/aixpert_nov93_PTX.html

AIX Versions 3.2 and 4 Performance Tuning Guide, SC23-2365

8.3 Performance Reporter

Performance Reporter is a program for systems management of RS/6000 and networks. It gathers systems management data, which are summarized and stored in a central database. Based on information from the database, Performance Reporter produces reports showing the summarized data.

Performance Reporter allows you to see recent details and trends over time for all important systems management areas. You can:

- Get a consolidated view of system and network performance across UNIX platforms (AIX, HP/UX and SunOS).
- See resource utilization to identify trends, bottlenecks, heavy usage, and idle resources.
- Detect out-of-line situations based on threshold values and predict problems rather than react to them.
- Have the statistics you need to set and report on service objectives.

8.3.1 Manager and Nodes

Performance Reporter uses one RS/6000 machine as the manager. The manager controls Performance Reporter operations, and it provides interfaces for administration and reports. The manager summarizes the data from the nodes and stores the information in a database. The lowest level of detail in the database is hourly data. Performance Reporter adds new data to the database every day, and the size of the database must be controlled so that it does not grow too large. The program keeps its database size under control by purging old data from the database each time a data collection is run.

You have to decide what kind of reporting you want to perform on each node in your network. Once this is done, using the manager interface, you activate data collection on each node. The data will be stored locally on the node in log files. If you want to make the same kind of reporting on several nodes, you can group them in a node group.

8.3.2 Operation

Once a day, the manager gathers data from each node it controls. This limits the effect on network traffic because data is transferred during off-peak hours. The information is summarized and then stored in files. Detailed data is retained for a short time. Monthly data are stored longer, according to storage policies that the user defined.

8.3.3 Reports

Using Performance Reporter's predefined reports, you can get management information covering the following areas. The information is based on data gathered through UNIX commands and from other products.

- Performance, based on UNIX commands (`iostat`, `vmstat`)
  - CPU statistics
  - Disk space statistics
  - Disk I/O statistics
  - Paging statistics
  - Availability

- Accounting, based on UNIX accounting:
  - Connect-time accounting
  - Process accounting
  - Disk-use accounting
  - Printer-use accounting
- Configuration, based on AIX commands:
  - Hardware (device) configuration
  - Software configuration
- Error analysis, based on the AIX error log
  - Error statistics
- Netview, based on SNMP MIB data:
  - Errors and warnings
  - Application alerts
- System Monitor, based on SNMP MIB data:
  - Session response times
  - User login times
  - Token ring traffic load, errors
  - Ethernet traffic load, errors

You can also modify these reports or create your own.

8.4 BEST/1

Today's RS/6000 servers run increasingly complex and large applications. Having tools that offer current usage statistics probably is not enough in today's fast-changing environments. This leads system managers to try to find some methods to anticipate performance problems. Unfortunately, this cannot be achieved by only using rules of thumb or linear projection. If such methods are applied to complex workloads, results are often inaccurate and meaningless. Using discrete simulation or workload benchmarks will yield good predictions, but their cost is generally high.

BEST/1, based on queuing theory, represents a good price/accuracy alternative. It can give IT managers valuable information on how to predict the behavior of their systems for a relatively low expense.

You can use BEST/1 to monitor performance and do capacity planning and trend analysis. This can be done on either a single machine (node) or in a C/S environment.

Capacity Planning

Capacity planning is the process of determining the system requirements for a defined workload in order to achieve a particular level of performance.

The main components of BEST/1 are: *Collect*, *Analyze* and *Predict.* These three modules will be discussed in detail in this chapter.

8.4.1 BEST/1 General Methodology

BEST/1 allows you to do capacity planning by collecting information on your server. By analyzing this data, you can identify main trends and components that characterize your server. Based on this information, you will be able to model the performance of your configuration. Once this step is done, you can start changing some elements of your model to observe the impact of this change. Typical changes are upgrading the processor, adding new users, adding or changing workloads, and changing disk configurations.

BEST/1 allows you to get and change information on:

- Hardware
  - Processor
  - Memory
  - Disk
  - Network
- Software
  - System
  - Process
  - Application (RDBMS: Oracle, Sybase)
- User and group

8.4.2 Getting Performance Data

Collecting Data: As the first step, *BEST/1 Collect* samples kernel structures and consults the object data manager (ODM) to gather information on resource utilization and system configuration. This is done by a "collector program" that will have minimal impact on the system. Performance statistics are collected at user specified intervals and summarized in order to give a complete overview.

Characterizing Workloads: *BEST/1 Analyze* allows you to take a closer look at the work being carried out by a system. This work is known as the workload. Within BEST/1, you can accept the standard workload (zzz), which contains all of the system activity, but it makes a lot more sense to create your own workloads. This enables you to analyze and make changes to individual system loads.

These workloads are made up of the following classes:

- User classes represent users having some relationships (same department, similar work).
- Transaction classes are composed of one or more processes that are related.

Note, although you can specify a single process in transaction classes, BEST/1 will automatically include any processes created from this.

Thus, a workload consists of processes carried out by users. The default workload contains all processes carried out by any user.

Creating workloads, user classes and transaction classes enables the user to accurately analyze and model a system, to produce meaningful reports on who and what are consuming system resources, for both real and modeled systems.

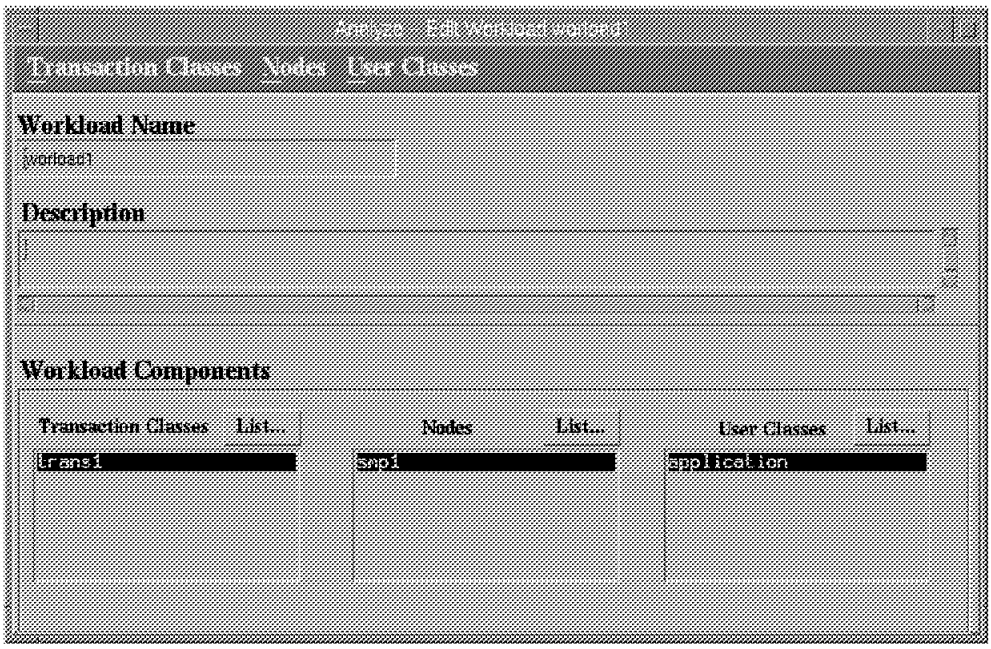


Figure 109. Example of Workload Definition

8.4.3 Capacity Planning with BEST/1

8.4.3.1 Predict Module

Once your model is created and evaluated, you can modify the following model components: *nodes*, *workloads*, *network*, *transactions*, *disks* and *logical volumes (LVs)*. With the *Predict* module, you can work interactively with all these components and see how configuration or workload changes affect the system performance.

BEST/1 Predict can also model entire domains of systems with multiple varied UNIX systems and calculate interactions and interferences between nodes. Independent and dependent transactions in C/S applications across nodes can be evaluated to calculate the domain-wide impact of changes.

For each component of the model, you can define two thresholds that will change the color of the pie chart in the main predict window (green, yellow and red). The lower threshold is called guideline the upper one threshold. Green means the value is below guideline; yellow means the value is between guideline and threshold; and red means it is above threshold. For nodes, networks and disks the threshold value reports percentage of utilization; for transactions and workloads response time is reported.

The predict main menu also reports general information on resource consumption and performance.

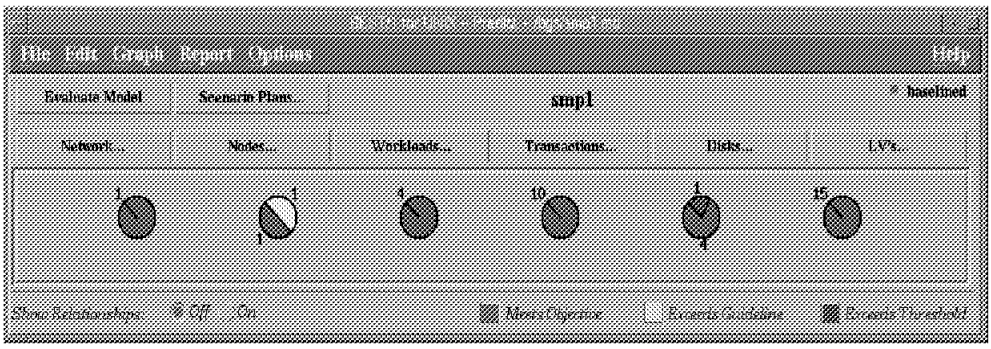


Figure 110. Main Predict Window

Nodes: Shows information on:

- CPU utilization
- System response time (average response time for transactions)
- Global throughput (transactions/hour)
- Queue length (number of transactions in the queue)
- Disks and logical volume configuration of the nodes

The user can modify the node data to test "what if" scenarios. This enables you to simulate a more powerful processor or even upgrade a uniprocessor to a multiprocessor.

Workloads: Displays information on the response time breakdown for each workload.

For workloads and transactions, the response time is decomposed into service time and wait time for the following components:

- CPU
- I/O
- Network
- Other (include, for example, the wait time for a transaction to run on another node)

By analyzing on which components the workload spends time, you can modify its characteristics to observe the impact on response time.

By modifying a workload, one can simulate new users, add new transactions, define new workloads, and see the impact of these changes on the overall system performance.

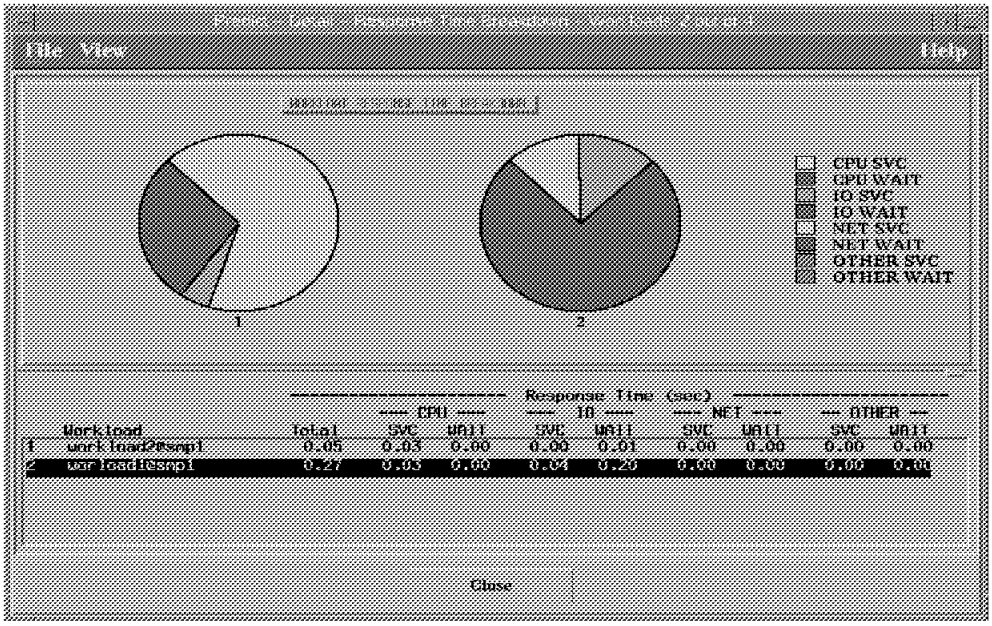


Figure 111. Example of Workload Response-Time Breakdown

Network: Gives information on the network your model is using. You can change some attributes, such as network type and network delay, to see the impact on workload response time.

BEST/1 uses only the traffic generated by your network adapter, not the utilization of your network. The *IBM NWAYS Campus Manager RMON* tools reveal the overall consumption of your network bandwidth.

Transactions: Displays various information such as response-time breakdown, throughput, and memory utilization.

You can modify transaction characteristics such as CPU time and synchronous or asynchronous I/O percentage and observe the changes on the model.

Disks: Displays information on the response time for a disk, the average utilization, and the I/O rate.

You can simulate new disk hardware and look at the effects on the system.

Logical Volumes: Displays information about response time and the I/O rate for each logical volume.

You can move or spread a logical volume across several physical disks and see how this modifies the behavior of your model. This is also useful if you want to balance I/O.

For both disks and logical volumes, you can use the *Edit I/O Subsystem* to graphically display the disk and logical volume configuration.

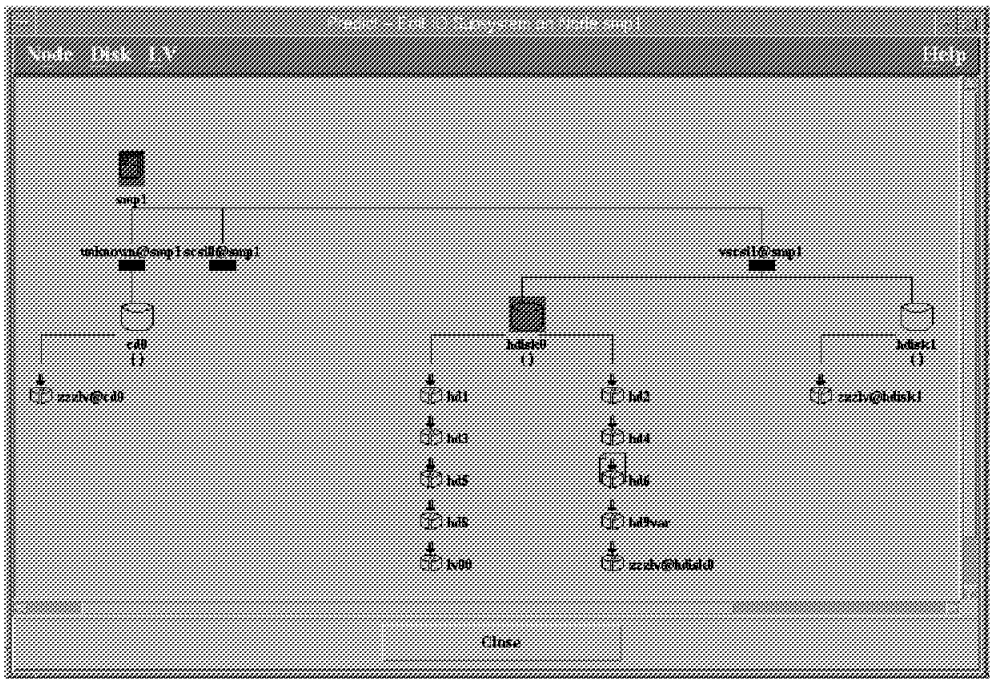


Figure 112. Example of Edit I/O Subsystem

8.4.3.2 Memory Modeling

AIX uses virtual memory to overcome the limitation of a relatively small amount of real memory. This virtual memory facility means that, as RAM becomes constrained, the system consumes more disk I/O because of paging activity. Additionally, AIX uses sophisticated features such as an VMM load control facility and a dynamic buffer for files. All this makes memory modeling difficult. BEST/1 provides useful information on how much memory is needed to avoid becoming constrained.

BEST/1 breaks down memory into four elements:

- Required System Memory: the amount of memory the kernel needs.
- Required Buffer Memory: other memory the system needs.
- Required Shared Memory: the total amount of shared memory the system needs.
- Required Private Memory: the total amount of all the resident set sizes (RSS) for each user process.

RSS

RSS tells how much RAM is currently used for the text and data segments for a particular process in units of kilobytes.

By adding all of these elements, BEST/1 reports the amount of RAM needed to achieve good performance.

8.4.4 Capacity Planning from Scratch

With BEST/1, you can save transaction classes into a library once you think it is accurate and representative.

If you need to design a new configuration, BEST/1 allows you to build trial models from scratch. You can create a new node with hardware characteristics (CPU, memory, disks, LVs, network adapter). From the library, you can add transactions and define users. By aggregating them, you are able to define workloads. Then, you can work with the predict module to simulate response time and configuration changes for performance prediction.

8.4.5 RDBMS Support

The current version of BEST/1 can collect and analyze RDBMS data (Oracle and Sybase). This allows you to get more-detailed information on how your database application is running and on statistics for RDBMS users and transactions.

8.4.6 BEST/1 and Benchmarks

BEST/1 reads a “hardware file” table to get performance characteristics of CPUs, disks and networks. You can modify this file and add your own hardware specification.

BEST/1 uses SPECint92 as its default performance rating. If you are mainly working with OLTP or RDBMS, it would be better to modify the “hardware file” and use TPC-C results or some other representative benchmark results. If your workload is mainly CPU intensive, you are better off using SPECint95 or SPECfp95. For more details on benchmarks, refer to Chapter 6, “Benchmarks” on page 149.

8.4.7 Other Capabilities of BEST/1

Here is a short presentation of other capabilities of BEST/1.

BEST/1 Monitor enables you to monitor, detect and analyze performance problems on nodes.

BEST/1 Manager allows you to plan automatic collect, analyze and predict runs. It also creates the file used by *BEST/1 Visualizer*.

BEST/1 Visualizer displays tabular or graph reports. It enables you to navigate from high-level information to in-depth details.

8.4.8 References

For more information on BEST/1:

BGS Systems, Inc.
128 Technology Center
Waltham, MA 02254-911

On the web: `http://www.bgs.com`
email: best1@bgs.com

For more information on IBM NWAYS Campus Manager RMON:

`http://www.raleigh.ibm.com`

8.5 NT Performance Monitor

NT workstation and NT server include a tool called the NT Performance Monitor, which can be used to gain insight on your system's performance and help you detect bottlenecks.

The NT Performance Monitor can help you detect bottlenecks locally or remotely by:

- Providing a view of resource usage (local and remote).
- Logging critical system values.
- Sending alerts when important events occur.

In the following sections, we give an overview of NT Performance Monitor characteristics.

8.5.1 Performance Monitor Terminology

The NT Performance Monitor tracks system resources, or *objects*, by assigning *counters* and *timers* to each resource to be tracked. An object's counter or timer records the activity level of the object. A counter is expressed as rate per second, while timers are expressed as the fraction of time that a device is used (shown as a percentage).

A unique set of counters and timers exists for the processor, memory, cache, hard disk, active processes and other object types that produce statistical information. This is the list of available objects:

- Browser
- Cache
- FTP Server
- HTTP Server
- Internet Information Services Global
- Logical Disk
- Memory
- NBT Connection
- NetBEUI
- NetBEUI Resources
- NWLink IPX
- NWLink NetBios
- NWLink SPX
- Objects
- Paging File
- Physical Disk
- Process
- Processor
- Redirector
- Server
- Server Work Queue
- System
- Telephony
- Threads

The objects are organized into a hierarchy. At the top of the hierarchy is the domain. Each domain contains computers. Each computer is further broken down into physical components such as processors, physical disks and memory. There are

other objects for logical or software components such as processes and the paging file.

8.5.2 Collecting Data with Performance Monitor

There are four possible views of data obtained with the performance monitor:

1. **Chart**: This view graphs counter values over time as the server is functioning. Within this view, you can select from graph and histogram formats.
2. **Report**: This view reports the counter values over time as the server is functioning in traditional ASCII format. The report view is especially helpful if you are trying to track many counters at the same time.
3. **Alert**: With this view, thresholds are set for each monitored counter. If the threshold is exceeded, an alert is logged.
4. **Log**: This view stores the captured data on the hard disk for later retrieval. This is useful when you do not have time to completely analyze the data in real time.

8.5.2.1 Chart View

As the counters are selected, this view graphs the values for the counters over time. You can chart many counters at one time. An example of a chart view is shown in Figure 113.

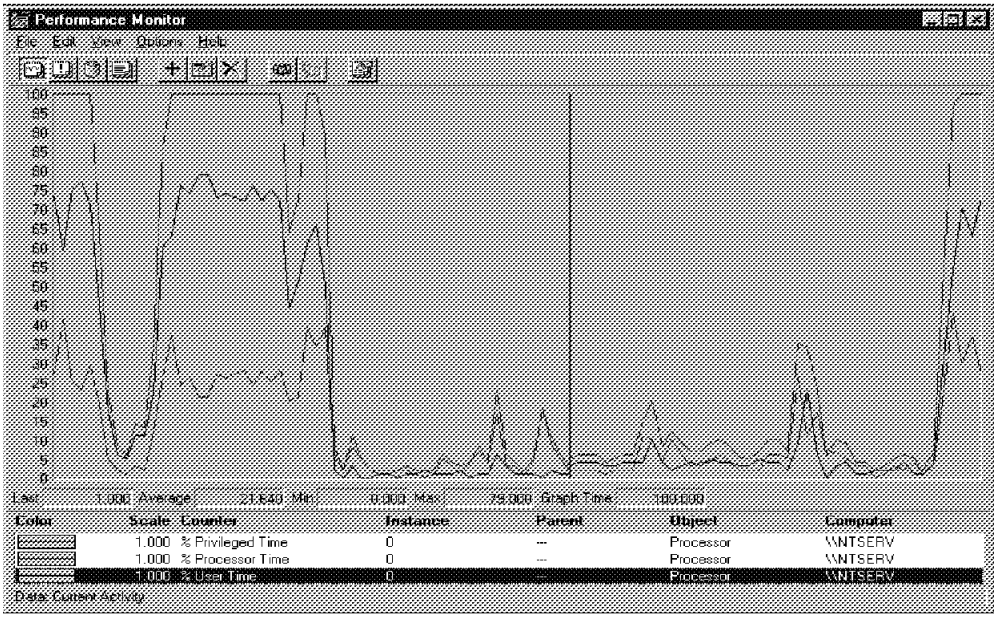


Figure 113. The Performance Monitor Chart View

The vertical line in the middle of the graph indicates the current point in time. It is always in red and occupies a space just beyond the last observed value. It moves to the right when the display is updated at the end of each time period. It wraps to the left edge of the chart at the end of the time period following the 100th data point plotted.

Disk performance counters are not ordinarily active because they typically reduce disk performance (less than 1 percent). Although any user can run Performance Monitor, only the administrator can turn on disk performance counters.

Note

To activate or deactivate disk performance counters, the `diskperf` command is used followed by a reboot of the system.

8.5.2.2 Report View

The report view is useful for monitoring many counters at the same time. This technique can be used to narrow down the number of variables that might be causing a problem. You can then view the problems more likely to occur using the chart view.

You can select multiple counters and multiple instances of the same counter. The counter values are displayed in a scrollable window, and they change in real time as the system is functioning. You can change the time interval over which the values are updated from the default of five seconds to whatever is appropriate for your problem.

An example of a report view is shown in Figure 114.

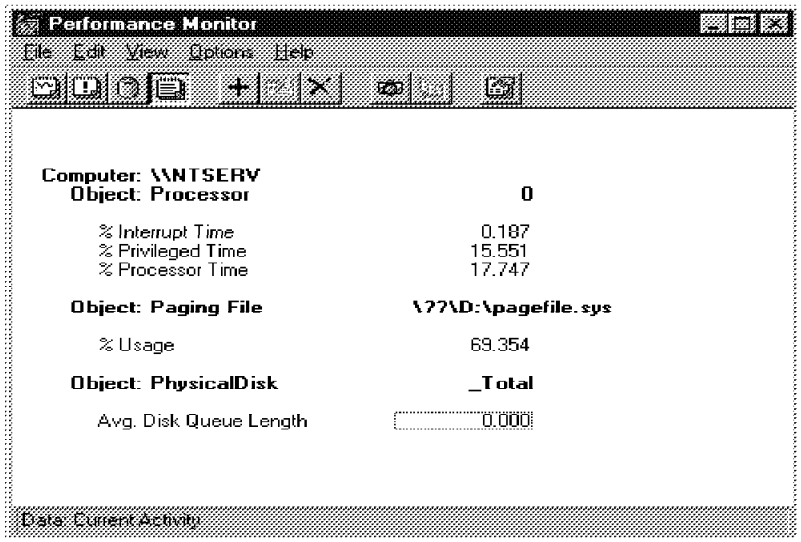


Figure 114. The Performance Monitor Report View

8.5.2.3 Alert View

This view is particularly useful if you want to see only exception conditions. An example of such a condition would be a low percentage of available disk space. With the alert view, you do not have to look through a tremendous amount of data to find what you are looking for. This type of monitoring can be most effectively used when the baseline measurements have been established and you want to only watch for conditions that you know could present a problem.

Figure 115 on page 294 shows an example screen from the alert view with some alerts already generated.

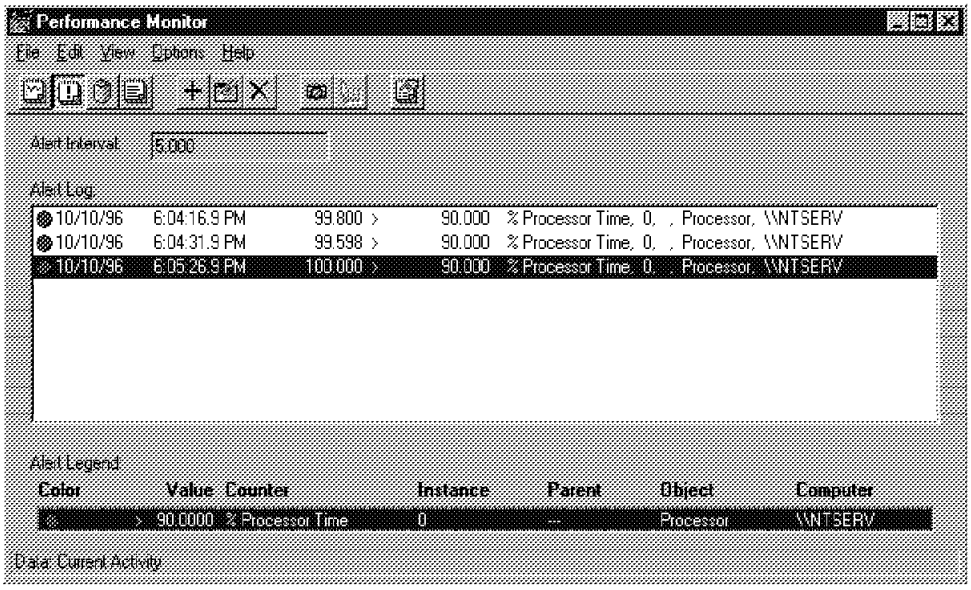


Figure 115. The Performance Monitor Alert View

8.5.2.4 Log View

To log captured data on monitored counters to a file, you will need to use the log view.

A panel is available for recording object activity. For each object logged, all instances of all counters will be logged. You cannot log individual counters or individual instances. Data can be logged in a new file, or, if you specify the name of an existing log file, new data will be appended at the end of file.

You can also play back data captured previously. Viewing data from a log file is very similar to viewing current activity: You can create charts, reports, alerts and even logs to a new log file, if desired.

While viewing logged data, you can move around within the log file and view data captured at different points in time.

8.5.2.5 Exporting Data

The `export` command on the file performance monitor main menu permits you to create either tab-separated or comma-separated ASCII files for import into other applications such as spreadsheets or word processors.

8.5.2.6 Printing Charts

NT Performance Monitor does not include a print facility for the charts you create. You can get printed charts by copying the chart into the clipboard and then pasting it into another tool, such as a word processor, and printing it as a document. Press **Alt+Print Screen** simultaneously to copy a chart into the clipboard.

8.5.3 Choosing What to Monitor

NT allows you to track a high number of system resources. You need to focus on the ones that are most critical to overall system performance. Fortunately, these are only a small subset of the possible resources available for monitoring. These are:

- CPU
- Disk subsystem
- System memory
- Network subsystem

Keeping a close watch on these and then taking the appropriate actions will help you maintain a well-tuned system. However, there are many more metrics that can be examined in the NT environment. For more information on monitoring NT system resources, please refer to the *NT Resource Kit, Volume 4: Optimizing Windows NT*. Almost the entire volume is devoted to monitoring system resources.

8.5.4 References

NT Resource Kit, Volume 4: Optimizing Windows NT

A Comparative Glance at AIX V4.2 and Windows NT 4.0, SG24-4784

Appendix A. Special Notices

This publication is intended to help system engineers, system administrators, and sales representatives understand the factors that determine the performance of applications running under AIX V4 on the IBM RS/6000. The information in this publication is not intended as the specification of any programming interfaces that are provided by IBM RS/6000 and AIX. See the PUBLICATIONS section of the IBM Programming Announcement for IBM RS/6000 and AIX for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (″vendor″) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

The following document contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples contain the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

400
AIX
AIX/6000
AIX PVMe
AT
CICS
CICS/6000
DB2
DB2/6000
ES/3090
ES/9000
ESCON
GXT150L
GXT150M
GXT1000
HACMP/6000
IBM
LoadLeveler
Micro Channel
NetView
OPC
Portmaster
POWER Architecture
POWER2 Architecture
POWERparallel
PowerPC
PowerPC 601
PowerPC 603
PowerPC 603e
PowerPC 604
RS/6000
RISC System/6000
S/370
SP
SP1
StorageSmart
System/370
SystemView
Ultimedia

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Microsoft, Windows, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

Java and HotJava are trademarks of Sun Microsystems, Inc.

ATM	Adobe Systems, Incorporated
XDF	Ametron, Incorporated
C++	American Telephone and Telegraph, Incorporated
NCS	Apollo Computer, Incorporated
Apple, Power Macintosh	Apple Computer, Incorporated
BEST/1, BGS	BGS Systems, Incorporated
Canon	Canon Kabushiki Kaisha
Compaq	Compaq Computer Corporation
CA	Computer Associates
CATIA	Dassault Systemes
Data General	Data General Corporation
Digital, VAX	Digital Equipment Corporation
Frame	Frame Technology, Incorporated
Gateway	Gateway Systems Corporation
HP/UX, Hewlett-Packard	Hewlett-Packard Company
POSIX	Institute of Electrical and Electronic Engineers
i950, Intel, Pentium	Intel Corporation
Always, Approach, Domino, Lotus, Lotus Notes, Notes, NotesBench, NotesMark, Replication	Lotus Development Corporation
PEX, X Window System, X Windows, X-Windows	Massachusetts Institute of Technology
NT, Windows 95, Windows NT	Microsoft Corporation
MIPS	MIPS Computer Systems, Incorporated
Motorola	Motorola Incorporated
NCR	NCR Corporation
Netscape	Netscape Communications Corporation
NT	Northern Telecom Limited
IPX	Novell, Incorporated
DCE, Motif, Open Software Foundation, OSF, OSF/Motif	Open Software Foundation, Incorporated
ODM	Optical Disk Mastering, Incorporated
Oracle, Oracle 7	Oracle Corporation
SCSI	Security Control Systems, Incorporated
OpenGL	Silicon Graphics, Incorporated
SPARCstation	SPARC International, Incorporated
SPECmark	SPECmark Standard Performance Evaluation Corporation
SPECfp92, SPECint92	Standard Performance Evaluation Corporation
NFS, Solaris, Sun, Sun Microsystems, SunOS	Sun Microsystems, Incorporated
Sybase, Sybase SQL Server	Sybase Corporation
SPEC	Systems Performance Evaluation Cooperative
TME, Tivoli, Tivoli Management Environment	Tivoli Systems, Inc., an IBM Company
TPC-A, TPC Benchmark	Transaction Processing Performance Council
AFS	Transarc Corporation
X/Open	X/Open Company Limited
Xerox	Xerox Corporation

Other trademarks are trademarks of their respective companies.

Appendix B. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

B.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How To Get ITSO Redbooks” on page 303.

- *A Comparative Glance at AIX V4.2 and Windows NT 4.0*, SG24-4784
- *Lotus Notes for AIX Capacity Planning, SG24-4844* (Available at a later date.)
- *Managing AIX V4 on PCI-Based RISC System/6000 Workstations (40P/43P)*, SG24-2581
- *A Technical Introduction to PCI-Based RS/6000 Servers*, SG24-4690
- *AIX/6000 X.25 LPP Cookbook*, SG24-4475
- *The IBM Xstation Handbook*, GG24-3695
- *An HACMP Cookbook*, SG24-4553
- *HACMP/6000 Customization Examples*, SG24-4498
- *Implementing High Availability on RISC/6000 SP*, SG24-4742
- *High Availability on the RISC System/6000 Family*, SG24-4551
- *DB2 Parallel Edition for AIX/6000 Concepts and Facilities*, SG24-2514
- *Backup, Recovery and Availability with DB2 Parallel Edition on RISC/6000 SP*, SG24-4695
- *RISC System/6000 Multimedia Environment: An AIX Ultimedia Services/6000 Overview*, GG24-4254

A complete list of International Technical Support Organization publications, known as redbooks, with a brief description of each, may be found in:

International Technical Support Organization Bibliography of Redbooks, GG24-3070.

B.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RISC System/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RISC System/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
Application Development Redbooks Collection	SBOF-7290	SK2T-8037
Personal Systems Redbooks Collection	SBOF-7250	SK2T-8042

B.3 Other Publications

These publications are also relevant as further information sources:

- *AIX Performance Tuning Guide*, *Versions 3.2 and 4,* SC23-2365
- *Performance Toolbox for AIX Guide and Reference, Version 1.2 and 2*, SC23-2625
- *AIX Version 4 System Management Guide: Operating System and Devices*, SC23-2525
- *IBM RISC System/6000 Technology*, SA23-2619
- *PowerPC and POWER2 Technical Aspects of the New IBM RISC System/6000*, SA23-2737
- *High Availability Cluster Multi-Processing 4.1 for AIX. Planning Guide*, SC23-2768
- *High Availability Cluster Multi-Processing 4.1 for AIX. Locking Applications*, SC23-2772
- *IBM Multimedia Server for AIX. General Information Release 1*, SC24-5798
- *IBM Multimedia Server for AIX. Guide and Reference Release 1*, SC24-5799

How To Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at URL http://www.redbooks.ibm.com.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **PUBORDER** — to order hardcopies in United States
- **GOPHER link to the Internet** - type GOPHER.WTSCPOK.ITSO.IBM.COM
- **Tools disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get lists of redbooks:

  ```
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET LISTSERV PACKAGE
  ```

  To register for information on workshops, residencies, and redbooks:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1996
  ```

  For a list of product area specialists in the ITSO:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Home Page on the World Wide Web**

  http://w3.itso.ibm.com/redbooks

- **IBM Direct Publications Catalog on the World Wide Web**

  http://www.elink.ibmlink.ibm.com/pbl/pbl

  IBM employees may obtain LIST3820s of redbooks from this page.

- **REDBOOKS category on INEWS**
- **Online** — send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL
- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** (Do not send credit card information over the Internet) — send orders to:

	IBMMAIL	**Internet**
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** — send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 Allerød
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** — send orders to:

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1) 415 855 43 29 (Outside USA)** — ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **Direct Services** - send note to softwareshop@vnet.ibm.com
- **On the World Wide Web**

Redbooks Home Page	http://www.redbooks.ibm.com
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank).

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number
- Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

DO NOT SEND CREDIT CARD INFORMATION OVER THE INTERNET.

List of Abbreviations

ACL	access control list
AFS	Andrew file system
AIM	Advanced Information Management
AIX	advanced interactive executive
ANSI	American National Standards Institute
API	application program interface
ASCII	American National Standard Code for Information Interchange
AT&T	American Telephone & Telegraph
ATM	asynchronous transfer mode
AUI	attachment unit interface
BCT	branch on count
bf	bigfoot
BIS	bus interface section
BIST	built-in self-test
BIU	bus interface unit
BMPX	block multiplexor channel
BNC	bayonet Niell-Concelman
BPU	branch processor unit
bps	bits per second
CA	continuous availability
CAD	computer aided design
CAM	computer aided manufacturing
CATIA	computer-graphics aided three-dimensional interactive application
CCITT	Comite Consultatif International Telegraphique et Telephonique
CDE	Common Desktop Environment
CD-ROM	compact disk read only memory
CDS	Cell Directory Server
CGI	Common Gateway Interface
CICS	customer information control system
CIM	computer integrated manufacturing
CISC	complex instruction set computer
CLIO/S	IBM Client Input/Output Sockets
COFF	common object file format
COP	common on-chip processor
CPI	cycles per instruction
CPU	central processing unit
CR	condition register
CRM	concurrent resource manager
C/S	client/server
CSMA/CD	carrier sense multiple access with collision detection
CTL	controller
CTR	count register
CTS	clear to send
DB	data base
DBMS	data base management system
D-Cache	data cache
DCB	data crossbar
DCD	data carrier detect
DCE	Distributed Computing Environment
DCE	data communication equipment
DCU	data cache unit
DFS	Distributed File System
DIX	name derived from principal developers of Ethernet (Digital Equipment Corporation, Intel, and Xerox)
DMA	direct memory access
DMP	distributed memory processor
DNS	domain name service
DP	double precision
DRAM	dynamic random access memory
DS	data stream
DSP	digital signal processor
DSR	data set ready
DSS	decision support system
DTE	data terminal equipment
DTR	data terminal ready
EAS	effective application speed-up
ECC	error correction code
EISA	extended industry standard architecture
ESCON	enterprise systems connection (architecture)
FC-AL	fiber channel arbitrated loop
FDDI	fiber distributed data interface
fdpr	feedback directed program restructuring
FFM	fraction in faster mode
FMA	floating-point multiply-add
FPR	floating-point register
FPU	floating-point unit
FTP	File Transfer Program

FXU fixed-point unit

F/W fast/wide

Gb gigabit

GB gigabyte

GDS global directory service

Geo-HA geographical high availability

GFLOPS giga floating-point operations per second

GIS geographic information system

GPC graphics performance characterization

GPR general purpose register

HA high availability

HACMP high availability cluster multi-processing

HAGEO High Availability Geographic

HAT hash anchor table

HFT high-function virtual terminal

HIPPI high performance parallel interface

HIPPI-PH high performance parallel interface. mechanical, electrical, and signalling protocol specification

HPC high performance computing

HP/UX Hewlett-Packard Co./UNIX

HTML Hypertext Markup Language

HTTP Hypertext Transfer Protocol

IBM International Business Machines Corporation

IBUS instruction dispatch bus

I-Cache instruction cache

ICU instruction cache unit

IDC International Data Corporation

IEEE Institute of Electrical and Electronics Engineers

IFU instruction fetch unit

I/O input/output

IOP I/O Processor

IP internet protocol

IPC instructions per cycle

IPC interprocess communication

IPX Internetwork Packet eXchange

IQDU instruction queue and dispatch unit

ISA industry standard architecture

ISDN integrated-services digital network

ISO International Organization for Standardization

ISV independent software vendor

IT information technology

ITSO International Technical Support Organization

IU integer unit

JFS journaled file system

JPEG Joint Photographic Experts Group

JTAG Joint Test Action Group

Kb kilobit

KB kilobyte

km kilometer

LADDIS Legato, Auspex, Digital, Data General, Interphase, Sun

LAN local area network

LFT low function terminal

LISP list processing

LPP licensed program Product

LR link register

LRU least recently used

LSU load/store unit

LU logical unit

LVM logical volume manager

LZ Lempel-Zev

M million

MAU multistation access unit

Mb megabit

MB megabyte

MC Micro Channel

MCA Micro Channel architecture

MCIU multiple-cycle integer unit

MFLOPS millions of floating-point operations per second

MHz megahertz

MIB management information base

MIPS millions of instructions per second

MJPEG motion JPEG

MMU memory management unit

MP multiprocessor

MPEG Moving Pictures Experts Group

MPI Message Passing Interface

MPL message passing library

MPP massively parallel processor

ms milliseconds

MTU maximum transmission unit

N/A not applicable

N/A not available

NAB name and address book

NAS numerical aerospace simulation

NBT NetBIOS over TCP/IP

NCS Netscape Commerce Server

NetBEUI NetBIOS extended user interface

NetBIOS network basic input output system

NFS network file system

NIC network interface card

NIS network information system

ns nanoseconds

NSAPI Netscape API

NT New Technology

NTS network terminal server

NTSC National Television Standards Committee

NTX network terminal accelerator

NW network

OC optical carrier

ODM object data manager

OEM original equipment manufacturer

OETP order entry transaction processing

OLTP on-line transaction processing

OPC OpenGL performance characterization

OpenGL open 3D graphics library interface

OPS Oracle Parallel Server

OS operating system

OSF Open Software Foundation Inc.

P2SC POWER2 Super Chip

PAL phase alternation by line

PA-RISC precision architecture RISC

PB petabyte

PBUS processor bus

PC personal computer

PCI peripheral component interconnect

PDT performance diagnostic tool

PE parallel edition

PEX PHIGS extension to X

PFT page frame table

PHIGS programmers hierarchical interactive graphics standard

PLB picture level benchmark program

PLL phase locked loop

POSIX portable operating system interface for UNIX

POWER performance optimization with enhanced RISC

PSDN packet switching data network

PSSP Parallel System Support Programs

PTF program temporary fix

PTPE Performance Toolbox Parallel Extensions

PTX Performance Toolbox

PVM Parallel Virtual Machine

PVMe Parallel Virtual Machine extended

RAID redundant array of independent disks

RAM random access memory

RAN remote asynchronous node

RDBMS relational database management system

RISC reduced instruction set computer

rmss reduced memory system simulator

RPC remote procedure call

rpm revolutions per minute

RS RISC system

RSC RISC single chip

Rsi Remote Statistics Interface

RSS resident set size

RTC real time clock

RTE remote terminal emulator

RTS ready to send

RxD receive data

SAP Systems, Applications, Products in Data Processing

SCIU single-cycle integer unit

SCSI small computer system interface

SCSI SE SCSI single ended

SCSI-2 F/W SCSI-2 fast/wide

SCU storage control unit

SDM System Development Multitasking

SE single ended

SECAM Sequentiel Coleur Avel Memoire

SF scaling factor

SFM speed-up of the faster mode

SFS shared file system

SG signal ground

SIO system I/O

SLIP serial line Internet protocol

SMDS switched multi-megabit data service

SMIT System Management Interface Tool

SMP symmetric multiprocessor

SNA system network architecture

SNMP simple network management protocol

SP Scalable POWERparallel

SP single precision

SPARC scalable processor architecture

SPEC Systems Performance Evaluation Corporation

SPMI system performance measurement interface

SPOF single point of failure

SPX sequenced packet exchange

SQL Structured Query Language

SRC system resource controller

SS SPARC Station

SSA serial storage architecture

stem scanning tunneling encapsulating microscope

STP shielded twisted pair

SU system unit

SunOS Sun operating system

SVC supervisor call

SVID system V interface definition

T1 transmission rate: 1.544 Mb/s

T3 transmission rate: 44.736 Mb/s

TB terabyte

TCP transmission control protocol

TLB translation lookaside buffer

TME Tivoli Management Environment

TP transaction processing

TPC Transaction Processing Council

TPM transactions per minute

TPP toward peak performance

TPS transactions per second

TTY Teletypewriter

TV television

TxD transmission data

UDP user datagram protocol

UP uniprocessor

UPS uninterruptible power supply

URL Universal Resource Locator

UTLB unified translation lookaside buffer

UTP unshielded twisted pair

VAX virtual address extension

VCR video cassette recorder

VFS virtual file systems

VHS video helical scan

VMM virtual memory manager

VSD virtual shared disk

WAN wide area network

X X Window System

XCOFF Extended Common Object File Format

XDF extended distance feature

XDR external data representation

XIO extended I/O

XOFF transmitter off

XON transmitter on

XPB X11 performance benchmark

XPC X performance characterization

XPG X/OPEN portability guide

XPS extended parallel server

YUV a color image encoding scheme that separates luminance (Y), red minus Y (U), and blue minus Y (V)

Index

Numerics

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Printed in U.S.A.