UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

COMPARACIÓN DE LA TECNOLOGÍA DE MICROPROCESADORES SUN UltraSPARC CON EL IBM POWERPC

INFORME DE SUFICIENCIA

PARA OPTAR EL TÍTULO PROFESIONAL:
INGENIERO ELECTRÓNICO

PRESENTADO POR:
JAIME RUBÉN ZAMBRANO CÓRDOVA

PROMOCIÓN
1992-I

LIMA – PERÚ
2007

COMPARACIÓN DE LA TECONOLOGÍA DE MICROPROCESADORES

SUN ULTRASPARC CON EL IBM POWERPC

a mi esposa por su apoyo incondicional
y a mis hijos de quienes esperamos
sigan nuestros pasos y nos superen.

SUMARIO

El presente trabajo, tiene por finalidad ser un documento de consulta para aquellos profesionales y estudiantes que requieran evaluar arquitecturas de dos importantes fabricantes de equipo de computo, los cuales hoy en día representan el avance en los diversos campos de investigación y desarrollo de la tecnología. Sabemos que cada día salen nuevos microprocesadores con mayor velocidad de reloj pero en muchos casos la diferencia esta basada en la arquitectura interna de dichos componentes, nuestro aporte es para aquellos que quieren conocer como diferenciar el rendimiento de un microprocesador no solo por la velocidad de reloj sino por la perfomance con carga real y en aplicaciones de uso comercial.

Así mismo le damos a conocer y diferenciar entre las diversas técnicas utilizadas para medir la performance ya sea con recursos propios o a través de mediciones con estándares internacionales, queremos darle al profesional o estudiante un panorama mayor para la evaluación de sistemas de procesamiento.

ÍNDICE

LISTADO DE FIGURAS Y TABLAS

PRÓLOGO

En el primer capítulo del documento, se tiene por finalidad describir genéricamente la arquitectura de un procesador y sus esquemas básicos de funcionamiento, describiremos los bloques funcionales de un procesador, la arquitectura de Von Neumann, la Unidad Central de Procesos (CPU), la memoria, las unidades de entrada y salida (I/O) y los buses del sistema.

En los Capítulos II al III, describimos los procesadores elegidos para este estudio: los procesadores que tienen mejores niveles de performance y que lideran las tablas de ranking mundial de evaluación a Mayo 2003.

Los procesadores elegidos fueron: SUN UltraSPARC IIIi y el IBM PowerPC 970 todos ellos de 64 bits.

El diseño e implementación de sistemas computacionales, involucra un gran número de soluciones de compromiso entre costo y desempeño de memoria, procesadores, subsistemas de Entrada / Salida, sistemas operativos, recursos de software, etc. Estas soluciones se reflejan en los estudios de selección y adquisición de equipos, software básico y aplicaciones.

Debido al alto costo y complejidad de estos sistemas, existe una necesidad de técnicas y herramientas para determinar las soluciones de equipamiento (software y hardware) más adecuadas de acuerdo a un cierto presupuesto. Esto también es necesario para la correcta planificación de la capacidad de los sistemas en operación.

En este contexto, las herramientas de evaluación del desempeño de sistemas computacionales apoyan al analista en la resolución de preguntas sobre el comportamiento del sistema en sus distintas fases del ciclo de vida, a decir:

Diseño e implementación de sistemas computacionales (Alternativas de equipamiento y de software, configuraciones, tiempos de respuesta y rendimiento esperados)

Sistemas en operación (evolución de la carga y configuración, incrementos en la carga, nuevas aplicaciones, subsistemas de discos, flexibilidad en el crecimiento, tanto en discos como en procesadores y memoria, impacto en el crecimiento de la red, etc.)

Medidas de Desempeño (Throughput, tiempos de respuesta promedios por cada tipo o clase de proceso - Batch, en línea, multiprogramación, multiusuario, interactivo, en tiempo compartido o en tiempo real usos de recursos por tipo de proceso y tiempos de respuesta esperado por cada tipo)

Métodos disponibles (Intuición y extrapolación de tendencias, evaluación experimental propia, contratada o públicas, simulación - modelamiento y análisis).

En el Capítulo IV se describen las herramientas de monitoreo que se utilizan en la actualidad para extraer los valores de los parámetros de intensidad de carga, de utilización de recursos y de rendimiento.

En el capítulo V se presentan las técnicas analíticas generales para la representación de los sistemas computacionales modernos y algunos conceptos básicos relacionados con las técnicas de pruebas de carga o "benchmarking".

Finalmente en las conclusiones damos las pautas de cómo utilizar los diferentes criterios para la selección del procesador que mas se ajuste a las necesidades específicas o generales, según sea el caso.

CAPÍTULO I
INTRODUCCIÓN

Dentro de la arquitectura de las computadoras existen varios bloques funcionales importantes los cuales en su conjunto conforman el Computador, a modo de una introducción simple pasaremos a describir dichos bloques para un mejor entendimiento.

1.1 Bloques funcionales del Computador Básico

Todo procesador tiene un bloque donde se almacenan las instrucciones a ejecutar, la Unidad Central de Proceso las ejecuta siguiendo el formato de cada instrucción y los resultados se llevan a la memoria de datos, visto desde un esquema realmente simplista.

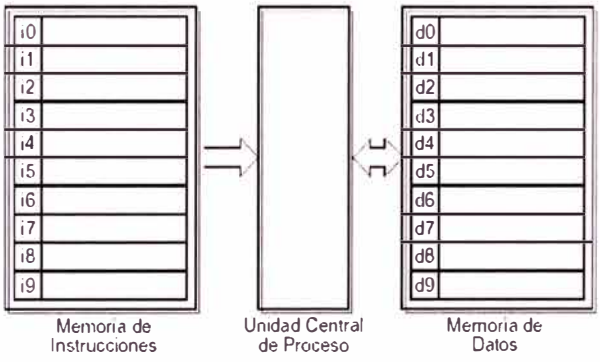


Figura 1.1 Bloques funcionales de un computador

La memoria de instrucciones tiene la lista de comandos en lenguaje de maquina que el procesador debe ejecutar; así en el ejemplo:

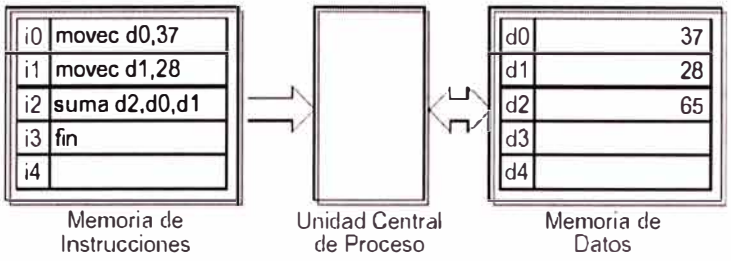


Figura 1.2 Ejecución de instrucciones de maquina

Luego de cargar 37 y 28 en las direcciones d0 y d1, en la instrucción i3 ejecuta la suma de esos valores guardando el resultado en d2.

1.2 Arquitectura de Von Neumann

La ENIAC (Electronic Numerical Integrator And Computer) fue la primera computadora electrónica de uso general en el mundo. Uno de los inconvenientes más grandes de la ENIAC era que tenia que ser programada manualmente mediante conmutadores y conectando y desconectando cables. El proceso de programación podría ser más fácil si el programa se representará en una forma adecuada para ser guardado en la memoria junto con los datos. Entonces, la computadora conseguiría sus instrucciones leyéndolas de la memoria, y se podría hacer o modificar un programa escribiendo en una zona de memoria.

Esta idea conocida como concepto de programa almacenado, se atribuye a los diseñadores de la ENIAC, sobre todo al matemático John Von Neumann. En 1946 Von Neumann y colegas empezaron el diseño de la nueva computadora que llamaron IAS y terminada hasta 1952, siendo el prototipo de toda una secuencia de computadoras de uso general. Salvo raras excepciones, todas las computadoras de hoy día **(1)** tienen la misma estructura general y funcionamiento que las máquinas de Von Neumann.

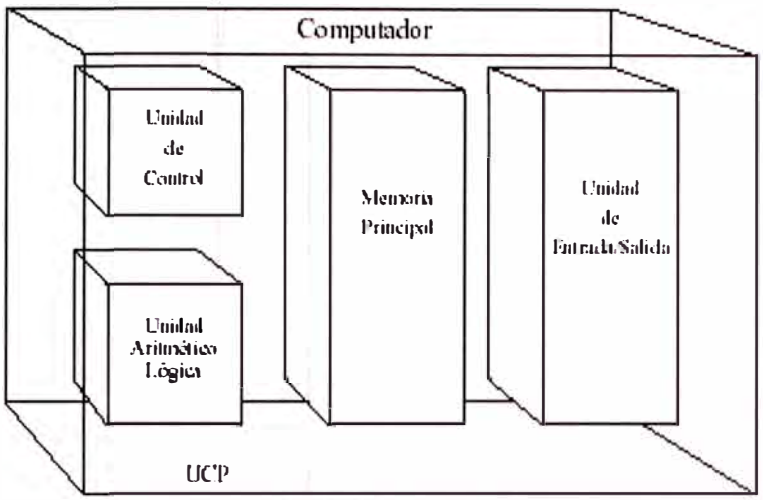


Figura 1.3 Ilustración Arquitectura de Von Neumann

1.3 La unidad central de proceso (CPU)

El procesador en términos funcionales es una caja negra que recibe como entrada instrucciones y datos, produciendo como salida nuevos datos.

Los datos son elaborados en su interior de acuerdo a un algoritmo expresado mediante las instrucciones. El procesador ejecutando las instrucciones secuencialmente genera como resultado los nuevos datos de salida.

Los datos de salida pueden ser ligeras modificaciones de los de entrada o incluso copias de los mismos, aunque normalmente la salida serán datos nuevos creados a partir de la entrada y modificados de acuerdo a complejos algoritmos. Nada impide que los datos de salida puedan ser la codificación de nuevas instrucciones para el procesador.

Finalmente, los datos de salida producidos por el procesador se almacenan en el sistema de memoria o se envían a los dispositivos periféricos que los necesiten y así lo requieran.

En un microprocesador de propósito general, la tarea a realizar se especifica en un programa.

Un programa consiste en una secuencia de instrucciones, codificadas (código máquina) de acuerdo a un formato interpretable por el procesador.

- Coordina el funcionamiento de todos los componentes de un computador.
- Originalmente se subdivide en:
    - Unidad de control (UC).
    - Unidad Aritmética Lógica (ALU).
- En los procesadores actuales:
    - Unidad de control (UC).
    - Ruta de datos (*Datapath*).

1.4 CPU multiprocesador

En ordenadores más grandes (*MainFrames*), la CPU puede estar formada por varios procesadores interconectados que interactúan entre sí para elaborar conjuntamente los datos de entrada y generar los datos de salida. Otra configuración posible es un procesador principal con microprocesadores específicos adicionales, que se dedican a realizar muy rápido trabajos parciales. De esta manera se descarga al procesador principal de parte del trabajo y se puede dedicar a las tareas más críticas como por ejemplo proceso de datos en tiempo real.

Estructuras más complicadas pueden ser necesarias para garantizar sistemas tolerantes a fallos, en los que son necesarios grupos ("clusters") de procesadores redundantes, para garantizar que un fallo en uno de los procesadores no afecte al proceso global que se este ejecutando.

Las posibilidades actuales de integración hacen viable que tanto la CPU como la memoria estén prácticamente incluidos en el mismo circuito integrado ("chip"). De este modo se acelera el tratamiento de datos incorporando internamente una memoria de acceso rápido (cache de nivel L1, e incluso el nivel siguiente, L2) comunicada directamente con el microprocesador. Por otro lado, han desaparecido elementos externos que realizaban parte del trabajo de cálculo de la CPU y que se podían encontrarse en ordenadores más antiguos, tales como los coprocesadores matemáticos. Estos componentes opcionales eran un complemento del microprocesador principal, y estaban especializados en realizar operaciones aritméticas con decimales. Actualmente ya están incluidos como una parte del procesador principal e integrado en el mismo circuito.

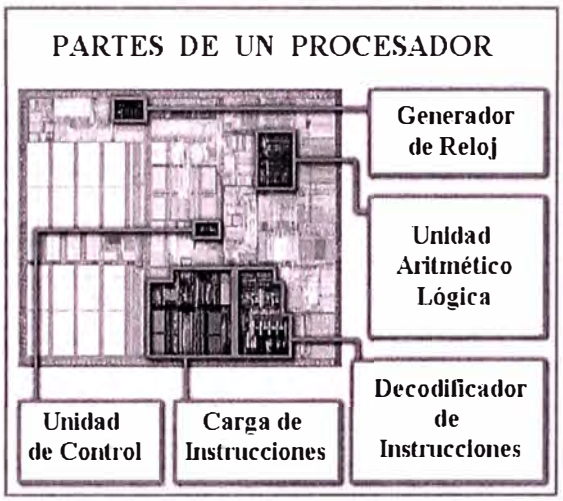


Figura 1.4 Las Partes de un procesador

1.5 Estructura de un microprocesador

En todo procesador se distinguen básicamente dos partes:

- **Unidad de Control:** encargada de realizar el control del proceso, es decir de generar las señales necesarias para activar los componentes de la unidad de tratamiento que actuarán sobre los datos en el instante de tiempo que corresponda. Para su funcionamiento suele disponer de los siguientes elementos. Decodificador de Instrucciones.: Unidad que interpreta el contenido del registro de instrucciones y permite generar las señales adecuadas para ejecutar la instrucción.
    - Decodificador de Direcciones: Unidad que interpreta la dirección en el registro de direcciones de Memoria MAR y selecciona la posición de memoria a ser accedida.
    - Registro Contador de Programa (PC). Guarda la dirección de la siguiente instrucción a ser ejecutada
    - Registro de Instrucciones (IR). Guarda la instrucción en ejecución

- **Unidad de Tratamiento o camino de datos**: agrupa a todos los componentes capaces de manipular los datos, es decir los recursos que disponemos en el interior del procesador.
    - Unidad Aritmética Lógica (ALU).
    - Registro Acumulador (ACC).
    - Registros de propósito general.

Estos componentes aparecen en todos los procesadores de propósito general. En los procesadores actuales pueden existir múltiples unidades de cada uno de los recursos para aumentar el grado de paralelismo al ejecutar un proceso, y de esta manera será posible ejecutar dos instrucciones simultáneamente.

Además también se incluyen unidades de cálculo más especializadas en el interior del procesador que no existían en los primeros procesadores de propósito general, tal como unidades de coma flotante FPU.

Por otro lado, existen procesadores que no son de propósito general y que llamaremos procesadores de propósito específico, los cuales solamente deben realizar una tarea concreta muy especializada. En estos casos es posible simplificar alguno de los componentes o incluso eliminar parte de ellos.

1.6 La Memoria

- Almacena tanto instrucciones como datos.
- Permite las siguientes operaciones:
  - Lectura.
  - Escritura.
- Se accede por dirección.
- No es el único medio de almacenamiento de un computador.

1.7 Las Unidades de Entrada / Salida

Conforman el Sistema de Entrada/ Salida del ordenador y son los encargados de proporcionar datos de entrada al procesador, así como de recibir los datos procesados para su comunicación al mundo exterior.

Permiten la comunicación entre el computador y el mundo exterior, y viceversa.

Esta formada por un conjunto de **puertos de E/S**. Pueden ser:

- de entrada (teclado)
- de salida (monitor)
- bidireccionales (disquetera)

1.8 Buses del sistema

Son conexiones que transmiten información de una parte a otra de la placa, se usa para hallar direcciones y/o datos. La evolución de los procesadores ha implicado el uso de buses con tamaños crecientes de 8 bits, de 16 bits, 32 bits y 64 bits.

Existen estándares diferentes: ISA, EISA, PCI y Microcanal. La más utilizada hoy en día es el bus PCI. Se pueden clasificar en:

- Bus de datos. Se utiliza para la transmisión de datos.
- Bus de direcciones. Se utiliza para indicar la dirección de memoria o el puerto de E/S al que se desea acceder.
- Bus de control. Se utiliza para coordinar el funcionamiento de todos los componentes de un computador.

CAPÍTULO II
PROCESADOR ULTRASPARC

2.1 Procesador UltraSPARC

Sun Microsystem desarrolla sus procesadores con la tecnología de Texas Instruments, esto le permite alcanzar una mejor tiempo de respuesta para múltiples aplicaciones de computo intenso tales como las siguientes:

- On-line transaction processing (OLTP)
- Media encoding and decoding
- Data Warehousing/Mining
- E-Commerce
- Messaging (E-mail), such as MS Exchange and Notes
- Graphics and image applications
- Networking
- Enterprise Resource Planning (ERP), such as SAP and People Soft

Figura 2.1 Chip de Microprocesador UltraSPARC IIIi

2.2 Familia de Procesadores UltraSPARC

Tabla 2.1 Tipos de sistemas utilizando el Procesador UltraSPARC

Tipos de Procesadores		
Nombre del Procesador	**Características**	**Desarrollado para:**
High-end Processors (s-Series)		
UltraSPARC III	• Escalabilidad hasta 106 procesadores. • Clase Enterprise (RAS). • 3rd generación de Procesadores de 64-bit UltraSPARC.	• Entry servers: V280 – V1280 (2-12 way). • High End workstations: Sun Blade 2000 (2 way). • NEBs servers: Netra 20 - 1280 (2 way-12 way). • Midframe servers Sun Fire 3800 - 6800 (8-24 way). • High End servers: Sun Fire 12-15K.
Mid-range Processors (i-Series)		
UltraSPARC IIIi	• Diseñado desde 1 a 4 procesadores. • Alto velocidad de procesamiento. • Alta integración de características y funcionalidades.	• Sun Fire V210 servers. • Sun Fire V240 servers.
UltraSPARC IIi	• Diseñado para sistemas de un solo procesador. • Bajo consumo de energía • Bajo costo. • Cache de nivel L2 de 512 KB integrado.	• Sun Fire V100 - V120 servers. • Sun Fire B100s blade server. • Netra CT 410 - 810 servers. • Sun Blade 150 workstation.
Entry-level Processors (e-Series)		
UltraSPARC IIe	• Diseñado para sistemas de un solo procesador. • Bajo consumo de energía • Bajo costo. • Cache de nivel L2 de 256 KB integrado.	• Sun Blade 100 workstations. • Sun Fire V100 workstations.

Para nuestro caso de estudio e interés hablaremos de la Familia UltraSPARC IIIi **(2)** dado que el objetivo es compararlo con otras fabricantes y queremos ser lo mas cercano posible en cuanto a dispositivos comerciales.

2.3 Descripción del Procesador

El procesador UltraSPARC IIIi consta de una arquitectura interna que le permite una rapidez en el procesamiento de información, en la figura 2.2 podemos apreciar la integración en un mismo chip de la memoria cache de nivel L1 para datos (64KB) y memoria cache L1 de instrucción (32KB), también contamos con memoria cache de nivel L2 (1MB), así mismo podemos apreciar tres unidades para el procesamiento de los datos.

El pipeline tiene 6 niveles de profundidad de operaciones(2 FP Units, 2 Integer Units, 1 Load/Store Unit y 1 Branch) todo esto ocurre en la ejecución de una instrucción.

Dichas unidades son activadas por la carga de información desde la memoria cache. Los datos y la información para la ejecución de las instrucciones son cargadas desde el Prefetch Cache (2KB) y el cache de datos (64KB), el cache de escritura (2KB) es usado para la salida de la información. Por último tenemos el controlador de memoria para el direccionamiento a la memoria RAM que permite la carga de los programas y los buses de comunicación (JBUS) con los dispositivos externos donde se graban, extraen, visualizan y almacenan los datos.

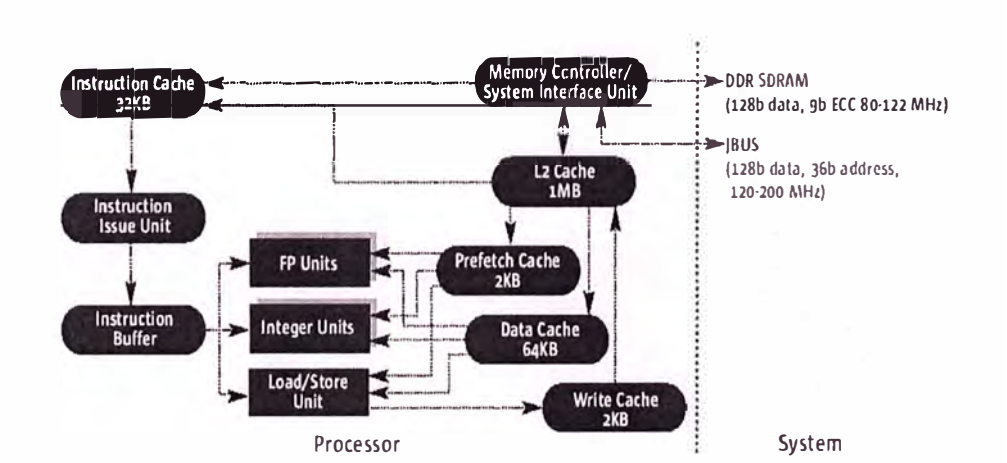


Figura 2.2 Arquitectura interna del Procesador UltraSPARC

2.4 Funcionamiento del Procesador UltraSPARC IIIi:

El UltraSPARC IIIi es un sistema SMP de 4 procesadores de 64bit cada uno, llega a 4MB en cache L2. Soporta un direccionamiento de 16GB de memoria por CPU, utiliza un sistema de memoria DDR-1 de 266Mhz, velocidad de acceso a memoria de 4.2GB/seg por procesador.

El punto más importante de este procesador es su sistema de bus para comunicaciones llamado JBUS **(3)** el cual opera un sistema tipo cache para los buses de I/O, dicho bus opera a 200MHz.

En el procesador UltraSPARC IIIi la tecnología desarrollada Multithreads permite realizar hasta 4 instrucciones por ciclo de reloj, lo cual da como resultado un mejor tiempo de ejecución de los programas y de los datos en las interfaces de I/O. CMT (Chip Multithreads). En la figura 2.3 podemos apreciar como se realiza dicha operación **(C=Computo; M=Latencia de Memoria)**

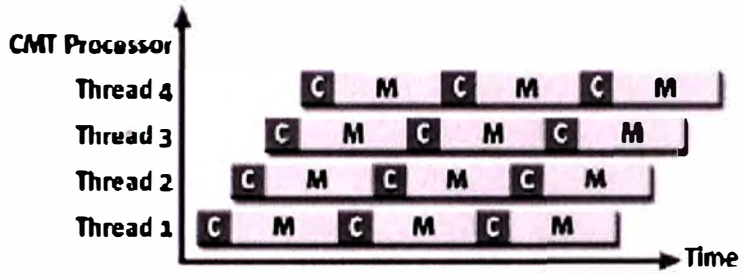


Figura 2.3 Single-core CMT Processor — Multiple Threads Compute During Memory Latency Times.

En la figura 2.4 podemos ver como se realiza la carga de las instrucciones desde la memoria cache integrada en el procesador.

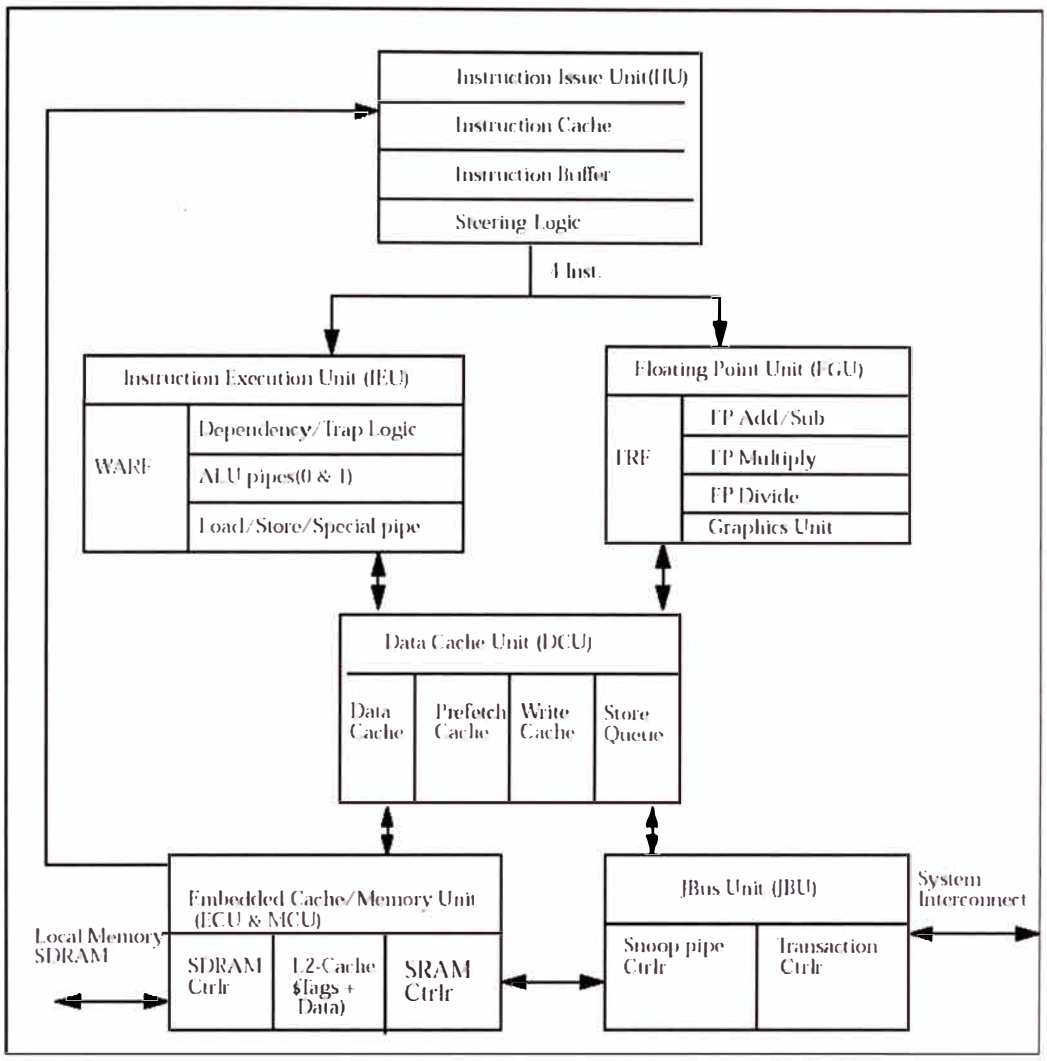


Figura 2.4 Diagrama de instrucciones del UltraSPARC IIIi

Para nuestro caso de estudio el Procesador UltraSPARC IIIi al contar con 4 procesadores puede llegar a realizar hasta 14 fases de acceso a memoria en un mismo ciclo de reloj fuera del tiempo de pipeline, así mismo trabaja con un reloj de 1.593 GHz, en la figura 2.5 podemos apreciar como se aprovecha dicho beneficio.

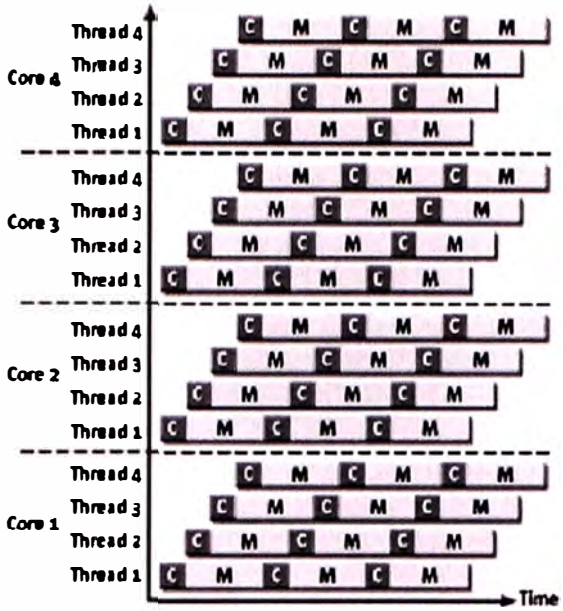


Figura 2.5 Multi-core CMT Processor — Executing Tens of Threads Maximizes Throughput.

2.5 Principales Características Tecnológicas de la Arquitectura

Tabla 2.2 Implementación de diseño del UltraSPARC IIIi

Parameter	UltraSPARC IIIi
Process	0.13µ CMOS
Frequency (GHz)	1.06
Metal Layers	6 copper 1 aluminum
Transistors (millions)	87.5M[2]
Core Voltage	1.4V
Peak Pwr. Dissipation (watts)	46@1.06GHz
Die Area (sq. mm.)	178.5 mm[2]
Pins	959
Package	ceramic PGA
L2 cache	1MB(4-way)

2.5.1 Arquitectura Externa del UltraSPARC IIIi 4-way SMP

En la figura 2.6 podemos apreciar como es el arreglo de los diversos CPU y las interfases de comunicaciones con los dispositivos de memoria RAM y los buses hacia los dispositivos externos I/O.

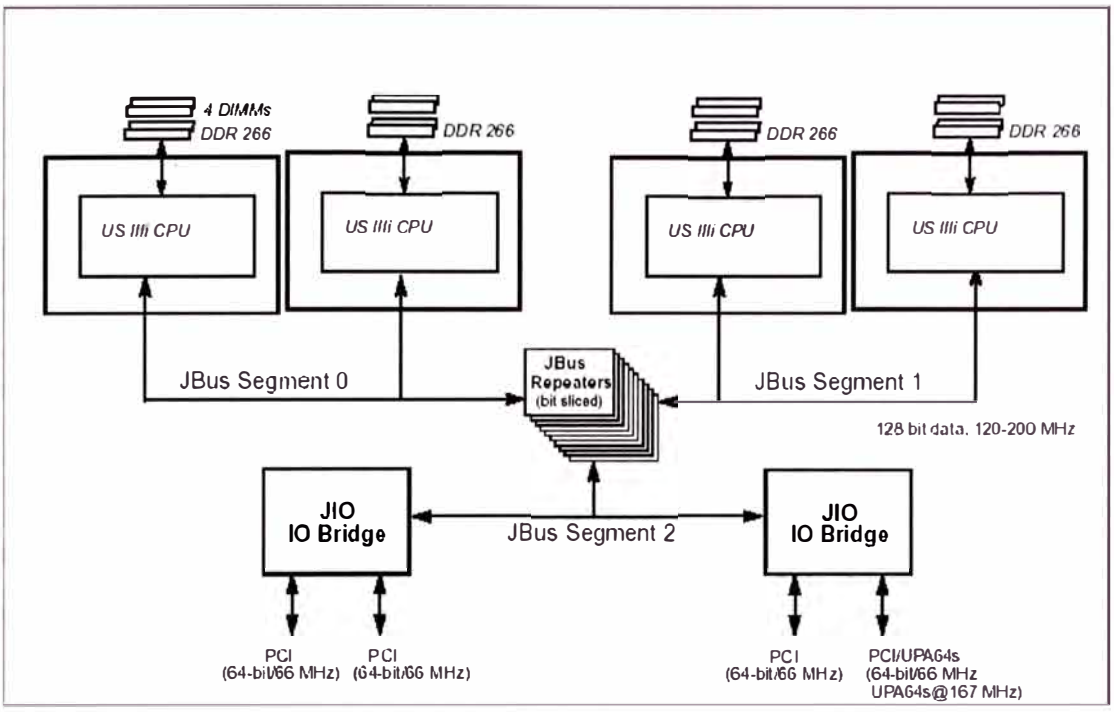


Figura 2.6 Arquitectura externa del UltraSPARC IIIi 4-way SMP

La Arquitectura externa del UltraSPARC IIIi esta conformada por 4 procesadores unidos a través de un arreglo de buses llamado JBUS, así mismo consta de un dispositivo integrado (Repeater) que permite la conexión con las interfaces internas y externas de la computadora buses I/O llamados en nuestro caso JIO.

2.5.2 Características del Jbus

• Bus de transacción compartido para datos y dirección de 16 byte (128bits)

• Diseñado para una frecuencia máxima de reloj de hasta 3 cargas de Jbus, una carga de Jbus puede ser a CPU, a un puerto JIO o a un puente (bridge)

• Velocidad de operación hasta 200 MHz

• Ancho de banda de 3.2 GB/s

• Chip de protocolo MOESI para un cache coherente que brinda alta performance y un gran ancho de banda para el sistema de multiprocesamiento

• Protocolo de distribución round-robin de arbitraje favorece el manejo del último puerto

• Bajo pin de conteo con 171 DTL (Dynamic Termination Logic)

• Soporte para orden de salida de datos con retorno a fuente

• Propagación de resultados simple punto a punto

• Flujo de control separado para direccionamiento y datos (encoded)

• Protección de paridad para Data y Control

• El JBus esta sincronizado con el reloj de distribución central

2.5.3 Características del JIO

• Integra gran ancho de banda, con baja latencia en las interfaces en un solo chip

• Utiliza bus estándar de la industria PCI I/O y tarjeta grafica Sun UPA64S

• Reduce el sistema complejidad / costo

• Interface con sistema de bus JBUS

• Soporta de 1 - 4 UltraSPARC IIIi-based sistema multiprocesador

• Generador de sistema de reseteo

• Compatibilidad binaria con Solaris y Drivers I/O

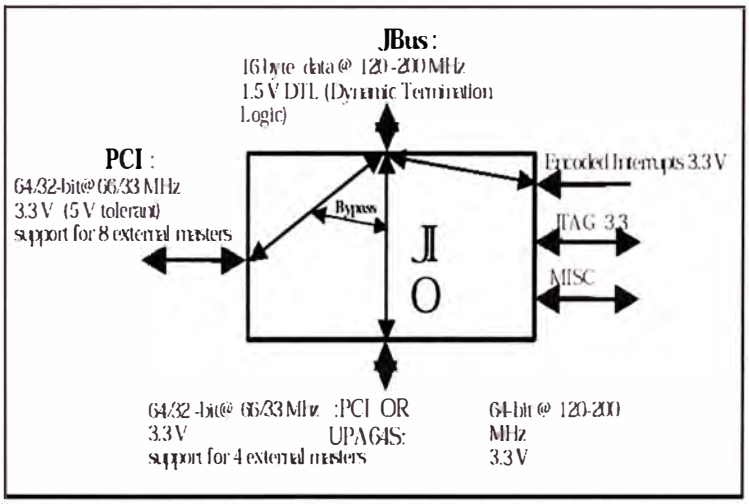


Figura 2.7 Vista del Flujo de JIO Interface / Data

2.5.4 Diagrama funcional de Bloques de las Interfases de Bus

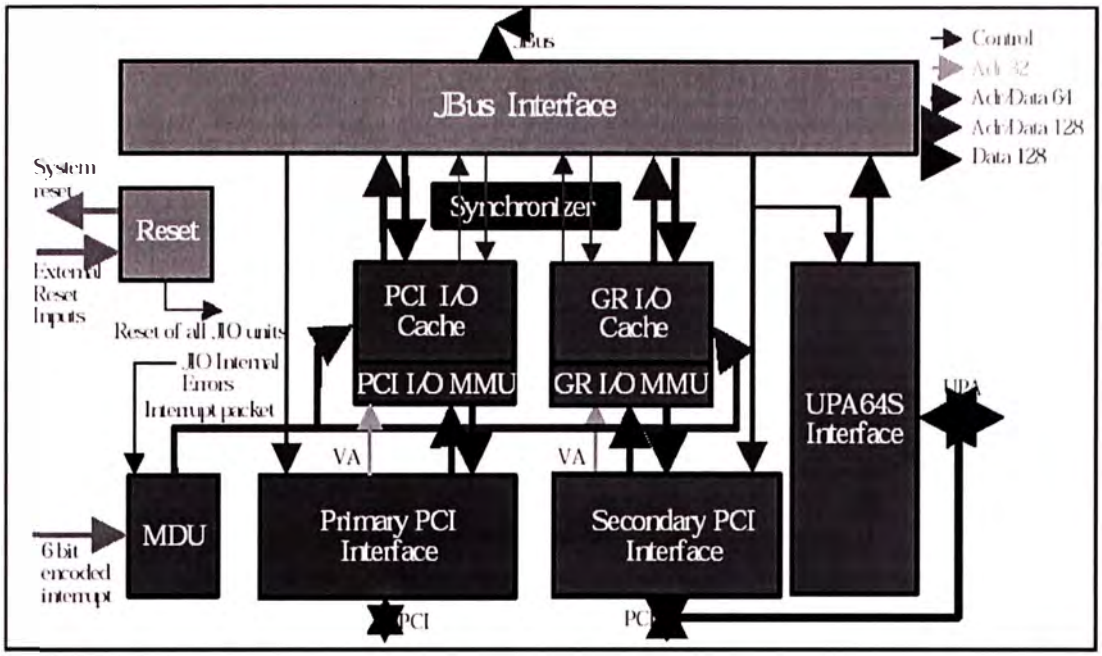


Figura 2.8 Diagrama de bloques del JBUS y el JIO

2.5.5 Detalles de las interfases de bus y la frecuencia de operación del reloj

Tabla 2.3 Detalle de interfases y frecuencia de reloj

System Interfaces	Clock Domains
JBus : 1.5 V DTL 16 byte bit shared address/data 120 – 200 MHz clock 179 signal pins	JBus PLL with PECL differential JBus system clocks as reference 100 – 200 MHz lock /operation range
Primary PCI : 3.3 v (5 v tolerant) LVTTL 64/32 bit shared address/data 25 – 66 MHz clock 103 signal pins	Primary PCI PLL with single ended full range swing PCI reference clock signal 25 – 66 MHz lock range 100 – 133 MHz range of operation
Secondary PCI/UPA64S Graphics : 3.3 v LVTTL 113 signal pins multiplexed PCI : 25 – 66 MHz clock UPA : 120 – 200 MHz clock	Graphics PLL which can be selectively configured as : PCI : Single ended full range swing PCI reference clock signal 25 – 66 MHz lock range 100 – 133 MHz range of operation UPA : PECL differential UPA system clock as reference 100 – 200 MHz lock/operation range

2.6 Unidad JBUS

• Provee las interfaces JIO's para el JBus

• Frecuencia operacional de hasta 200 MHz

• Respuesta al CPU con reducción de la latencia en 3 ciclos de carga del JBus para todos los casos menos los invalidados

• Mantenimiento de una copia cache de I/O tags

• Arbitraje punto a punto, solo en el nivel JBus

• Soporte de hasta 4 lecturas externas por cache

2.7 Unidad de Cache I/O

• Sirve como buffer de prefetch coherente con la memoria principal

• Fully set associative cache memory of 8 lines of 64 bytes

• Frecuencia máxima operacional de 133 Mhz

• Uses Read-Modify-Write / Write-back write policy for < 64 byte writes

• Puente cache para memoria de 64 byte (cache line) de escritura

• FIFO-like counter-based replacement policy

• CSR-enabled prefetch of 1,2,3 cache lines for PCI commands

• CSR-enabled prefetch stride of 1 - 127 cache lines within 8K page

• Prefetch both on a hit or miss

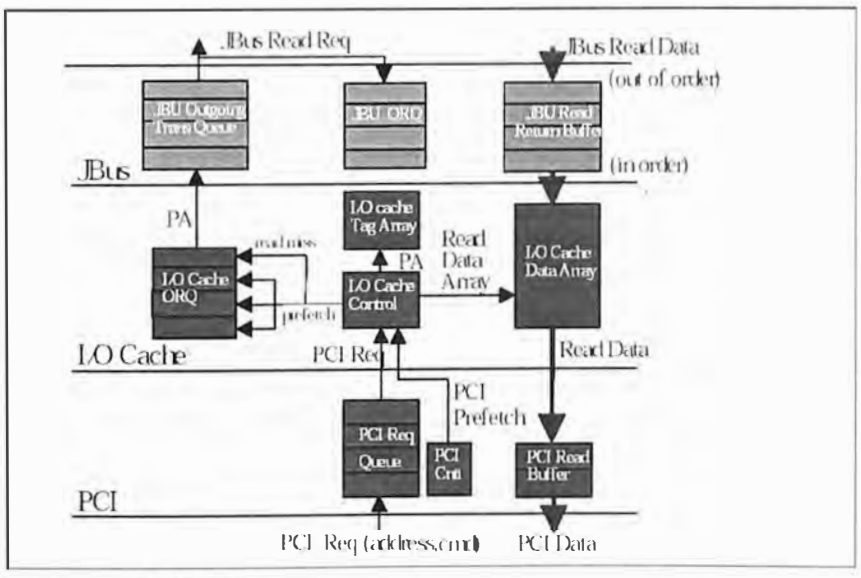


Figura 2.9 Cache I/O y PCI Prefetch Pipe

2.8 Unidad PCI

• Provee las interfaces de JIO's estándar de la industria PCI bus (2.2 spec)
• Levantamiento Interno de transacciones DMA en 64 byte (línea cache) limite
• Dos niveles bajos y uno alto de prioridad alrededor del bus PCI
• CSR - prefetch disponible de 1 línea de cache para comandos PCI
• Dos caches de línea profunda para buffer de datos mejorados DMA de lectura

2.9 Unidad de interface UPA

• Provee interfaces JIO's para Sun sistema de bus UPA64
• Arquitectura mejorada de Puerto unificado Sun 2.0
• Solo una interface principal
• Velocidad de Operación de 100 a 200 Mhz (UPA64S clock domain)
• Máximas transacciones externas de un total 16 total
• Hasta 4 lecturas y 12 escrituras
• Hasta 16 escrituras (no lecturas)

2.10 Especificaciones Técnicas

2.10.1 Performance de JIO

La performance del bus I/O es de 2 GB/s (con 2 JIO chips)

Tabla 2.4 Performance del DMA

PCI DMA, 64-bit @ 66 MHz, 1 master

Burst Size (bytes)	DMA write (MB/s)	DMA read (MB/s)
32	301	162
64	384	248
128	444	281
1024	515	465
8192	515	465

Lectura y escrituras ininterrumpida desde tarjetas PCI (64-bit/66 MHz), performance Pico PCI para pequeños quiebres (32 or 64 bytes).

Tabla 2.5 Performance del PIO

PCI Memory PIO, 64-bit @ 66 MHz

Burst Size (bytes)	PIO write (MB/s)	PIO read (MB/s)
4	88	62
8	106	106
16	211	211
64	384	384

Tabla 2.6 Performance del UPA

UPA Store Throughput, JBus @ 200 MHz

Burst Size (bytes)	UPA Bus Frequency		
	200 MHz (MB/s)	150 MHz (MB/s)	100 MHz (MB/s)
8	**1.55**	**1.12**	**0.80**
16	**1.50**	**1.20**	**0.80**
64	**1.32**	**1.12**	**0.80**

2.10.2 Datos de implementación del Bus JIO

- Puerto de conteo ASIC : 922 K
- Almacenamiento de Memoria : RA - 55 K, Flops – 29 K
- Tamaño : 9.5 mm. sq.
- Pin de señales : 433
- Máximo Consumo de energía : 10 Watts
- Tecnología : 0.25 Micrón CMOS

2.11 Costo de Servidores con Tecnología Sun UltraSPARC IIIi

Para tener un estimado de lo que puede valer un computador con procesador UltraSPARC IIIi adjuntamos un cuadro comparativo de la marca SUN **(4)**:

Tabla 2.7 Costo de diferentes Servidores UltraSparc IIIi

Select a Base Configuration				
	SMALL	MEDIUM	LARGE	EXTRA LARGE
UltraSPARC IIIi Processor	2 @ 1. 593 GHz	4 @ 1. 593 GHz	4 @ 1.593 GHz	4 @ 1.593 GHz
Internal Cache per Processor	1 MB	1 MB	1 MB	1 MB
Memory	4 GB (8 @ 512-MB DIMMS)	8 GB (16 @ 512-MB DIMMS)	16 GB (16 @ 1-GB DIMMS)	32 GB (16 @ 2-GB DIMMS)
10000 RPM Ultra320 SCSI Disk Drive	4 @ 73 GB	4 @ 73 GB	4 @ 73 GB	4 @ 146 GB
DVD-ROM Drive	1	1	1	1
10/100/1000 Mb/s Ethernet Port	2	2	2	2
Serial Port	1 - DB9	1 - DB9	1 - DB9	1 - DB9
USB Port	4	4	4	4
Power Supply	2 (1 + 1)	2 (1 + 1)	2 (1 + 1)	2 (1 + 1)
Enterprise Infrastructure Software	Java Enterprise System Pre-Installed	Java Enterprise System Pre-Installed	Java Enterprise System Pre-Installed	Java Enterprise System Pre-Installed
Operating System	Solaris 10 Pre-Installed	Solaris 10 Pre-Installed	Solaris 10 Pre-Installed	Solaris 10 Pre-Installed
List Price	$13,995.00	$23,995.00	$30,995.00	$40,995.00

2.12 Set de Instrucciones (VIS)

Es un conjunto de instrucciones que facilita la integración del SIMD (single instruction Multiple Data) y la unidad de punto flotante (FPU). Sun provee un set completo de más de 80 instrucciones las cuales pueden ser categorizadas a continuación:

- Instrucciones Aritméticas y Lógicas
- Instrucciones de Comparación
- Instrucciones de conversión de formatos
- Instrucciones de ayuda para data desalineada
- Instrucciones de Acceso a memoria
- Instrucciones de Arreglo para datos tridimensional
- Instrucción especifica para aceleración de video MPEG
- Instrucción de manipulación flexible de data

Para hacer uso de las instrucciones debemos de considerar el manejo de las variables como se indica en la figura adjunta:

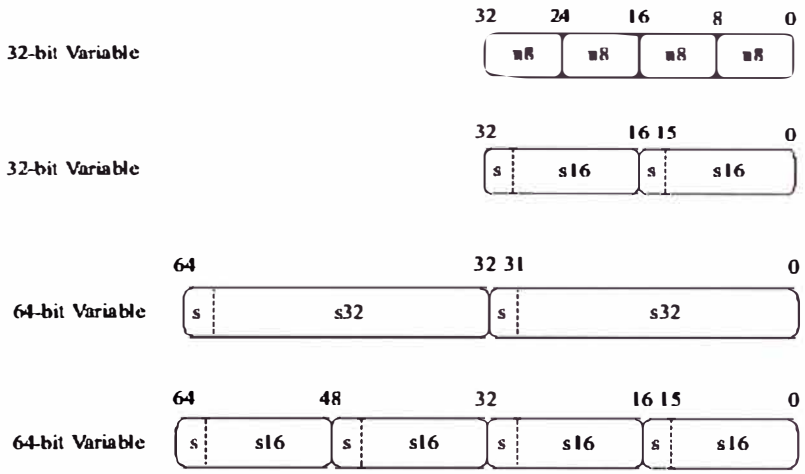


Figura 2.10 Tipos de Datos

Instrucciones Aritméticas y Lógicas.- provee la facilidad para adicionar, sustraer y multiplicar hasta 4 diferentes variables en una sola operación. Así mismo permite realizar instrucciones lógicas (and, or, xor) en paralelo mientras se realizan instrucciones de punto flotante.

Algunos ejemplos:

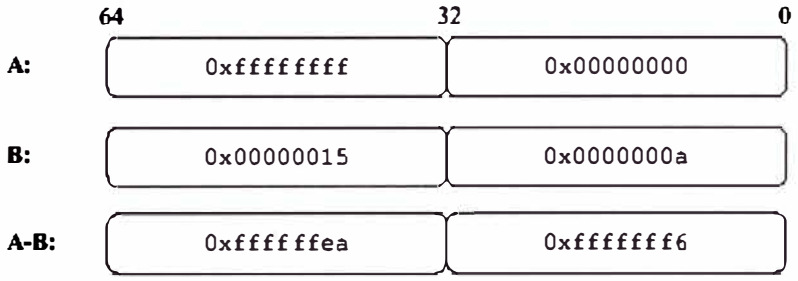


Figura 2.11 Instrucción de sustracción

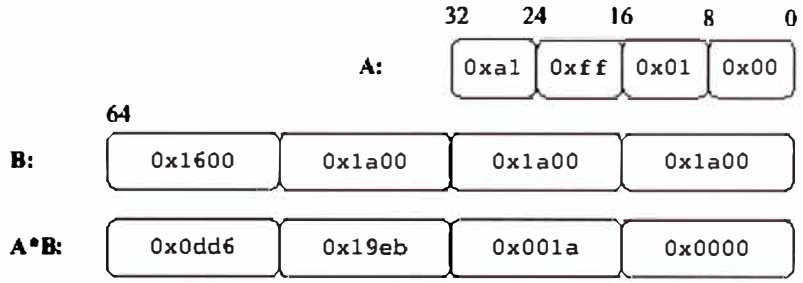


Figura 2.12 Instrucción de multiplicación

Instrucciones de Comparación.- provee la facilidad de hacer comparaciones sobre variables de 64bits con particiones de 16 o 32 bits. Las funciones de less-than, less than or equal to, greater-than, greater than or equal to, equal to y not equal to. El resultado de la comparación se muestra en 2 o 4 bits contenidos en el orden más bajo de una variables de 32 bits.

Para el caso de comparación mayor que o menor que el 1 significa verdadero, para el caso de igualdad el 1 significa verdadero el cero significa falso.

Algunos ejemplos:

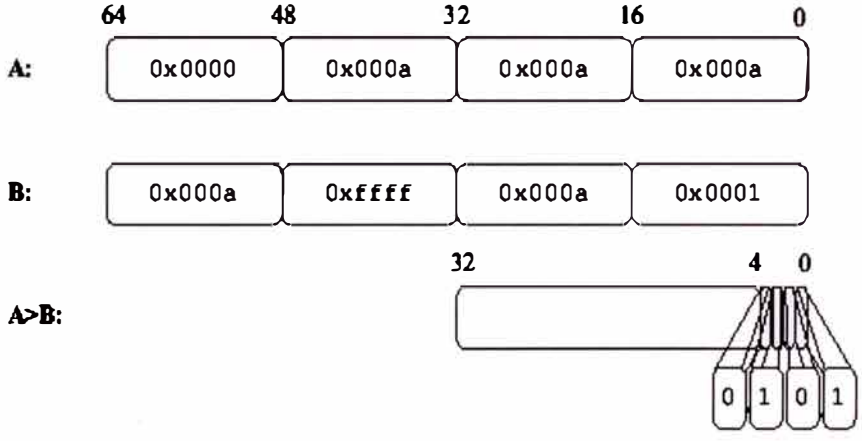


Figura 2.13 Instrucción de comparación mayor que

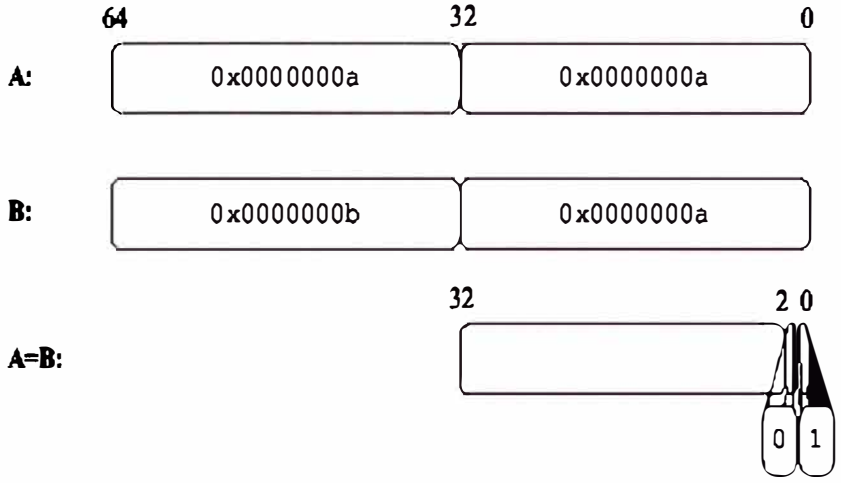


Figura 2.14 Instrucción de comparación igual que

Instrucciones de Conversión de formato.- permite la conversión de datos para facilitar carga de nivel paralelo y de alcanzar un mejor grado de precisión. El set de instrucciones provee 32-8, 32-16 y 16-8 bit de conversión.

La instrucción usada más frecuente es **fpmerge**, la cual genera un resultado de 64bits de una combinación alterna de A y B bytes.

A continuación se muestra dicha instrucción:

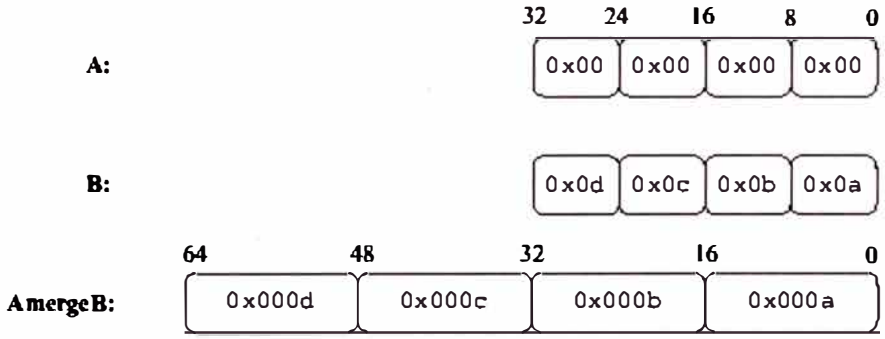


Figura 2.15 Instrucción de conversión fpmerge

Instrucción de alineamiento de datos.- provee la facilidad de concatenar datos en una dirección de memoria, permitiendo que estos ocupen espacios de memoria continua, el armado en bloques de distintas variables de datos hace que se utilicen instrucciones como la **faligndata** que permite alinear los datos en memoria.

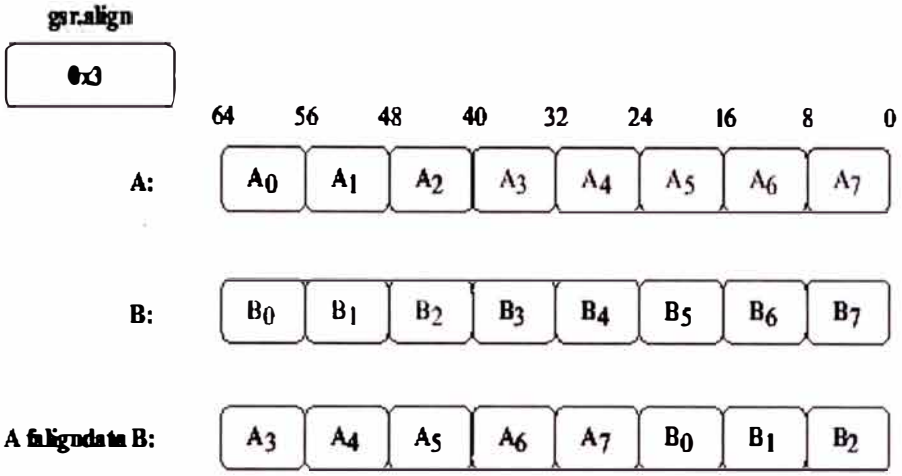


Figura 2.16 Instrucción de alineamiento faligndata

Instrucción de acceso a data.- permite recuperar información almacenada en memoria de forma alineada, la recuperación de dicha información se realiza mediante la instrucción **edge** , dicha instrucción permite rápidamente recuperar la data almacenada en forma parcial.

La carga de bloques y de instrucciones de almacenamiento vienen siendo utilizadas con éxito para acelerar funciones como **memcpy.**

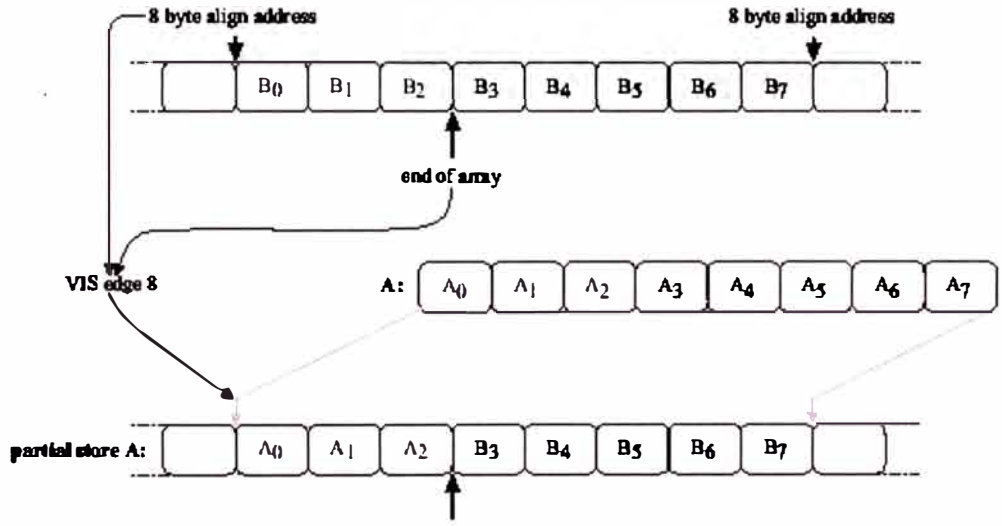
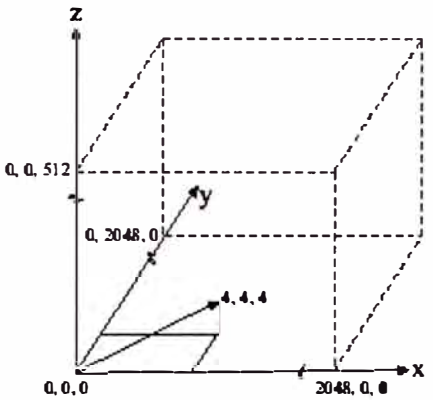


Figura 2.17 Instrucción de acceso a data

Instrucciones para arreglo de datos tridimensionales.- provee facilidad de almacenamiento de información tridimensional cuando esta requiere algún cambio en alguna coordenada, evitando perdida de memoria cache, utilizando el formato de transformación de estructura se consigue mejorar el espacio de almacenamiento, evitando ocupara y algunas veces perder información.



Voxel address	Standard offset	VIS offset
1, 1, 1	8MB	42B
2, 2, 2	16MB	16.01KB
3, 3, 3	24MB	16.06KB
4, 4, 4	32MB	33.06KB

Figura 2.18 Instrucción de arreglo tridimensional

Instrucción especifica para aceleración de video.- provee el algoritmo para precisar si una información esta cambiando de estado, esto facilita en aplicaciones de video MPEG solo cambios mínimos en el comportamiento de la señal. La instrucción **pdist** permite obtener el valor absoluto entre los ocho pares de bytes (pixels) contenidos en A y B, la suma de ese resultado (i) es comparado con la del intervalo anterior (i-1).

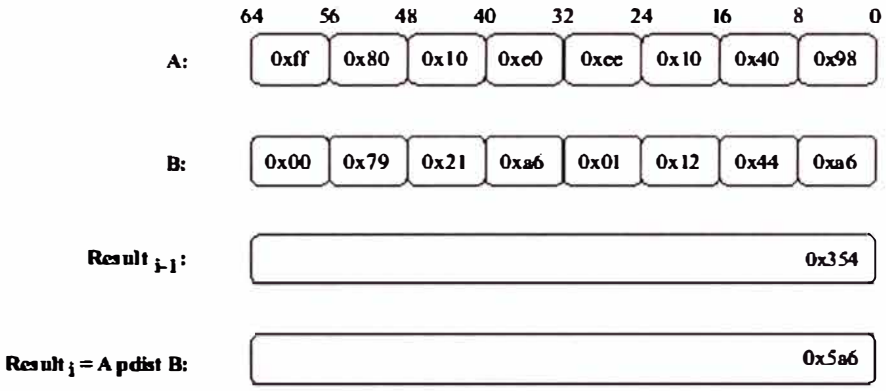


Figura 2.19 Instrucción de aceleración de video

Instrucción de manipulación de data.- provee las instrucciones para combinar dos variables de 64 bits cada una, para ello se utilizar la instrucción **bshuffle** , en combinación con registro global dedicado (**gsr** registro) el cual indica el orden en que se debe manipular ambas variables.

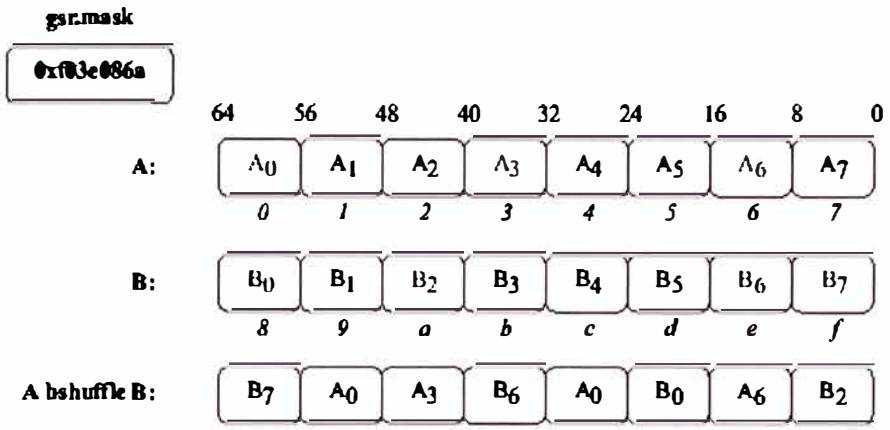


Figura 2.20 Instrucción de manipulación de data

2.13 Librería de software

Sun provee una variedad de macros y software abstracto para que las rutinas de operación sean fácilmente implementadas con los procesadores UltraSPARC, dicha librería **(5)** cuenta con más 3,000 funciones las cuales permiten acelerar el tiempo de proceso permitiendo mejorar de entre 1.3 y 10 veces el ciclo de proceso. Esto a su vez permite facilitar la migración desde otras plataformas las aplicaciones que son necesarias.

Tabla 2.8 Funcionalidad de la librería de software

mediaLib Software Components					
Algebra	Graphics	Image	Signal	Video	Volume
Matrix operations	Draw and fill circles and ellipses	Basic Arithmetic: add, ave., blend	Basic DSP: FFT, FIR, IIR	DCT/IDCT	Maximum intensity searching
Vector operations	Draw and fill polygons	Convolutionand auto correlation	ADPCM	Motion estimation	Ray casting functions
Data manipulation operations	Draw lines and arcs	Max, min, thresholding	Windowing	Color conversion	
		Rotate and zoom	Convolution and correlation	Interpolation	

CAPÍTULO III
PROCESADORES POWERPC

3.1 Procesador PowerPC

PowerPC es una familia de procesadores fabricados por IBM y Motorola, para satisfacer las demandas del mercado computacional, orientados a desktops y servidores de propósito general, entre los principales fabricantes de estaciones de trabajo y servidores están IBM y Macintosh. El diseño fue basado en su exitoso microprocesador Power4 del cual usaron su tecnología y le adicionaron mejoras en velocidad, bajo consumo de energía y multiprocesamiento.

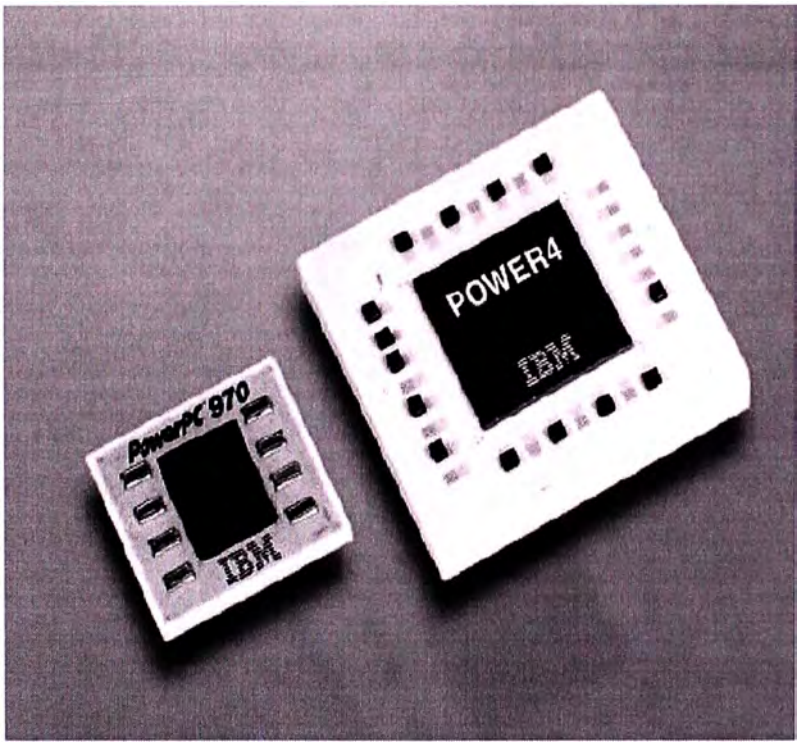


Figura 3.1 Tamaño del Procesador IBM PowerPC 970 vs Power4

3.2 Familia del Procesador PowerPC

IBM entrega 3 modelos o tipos de procesadores PowerPC **(6)**.

- **La serie 9xx**, la cual es un microprocesador de 64bit, que brinda alta performance a diversas aplicaciones que requieren gran demanda de computo.
- **La serie 7xx**, la cual es un microprocesador de 32bit, que brinda alta velocidad de procesamiento con un bajo consumo de energía.
- **La serie 4xx**, la cual es un microprocesador que brinda un sistema de 32 bits en el procesador, incrementa la funcionalidad en un solo chip.

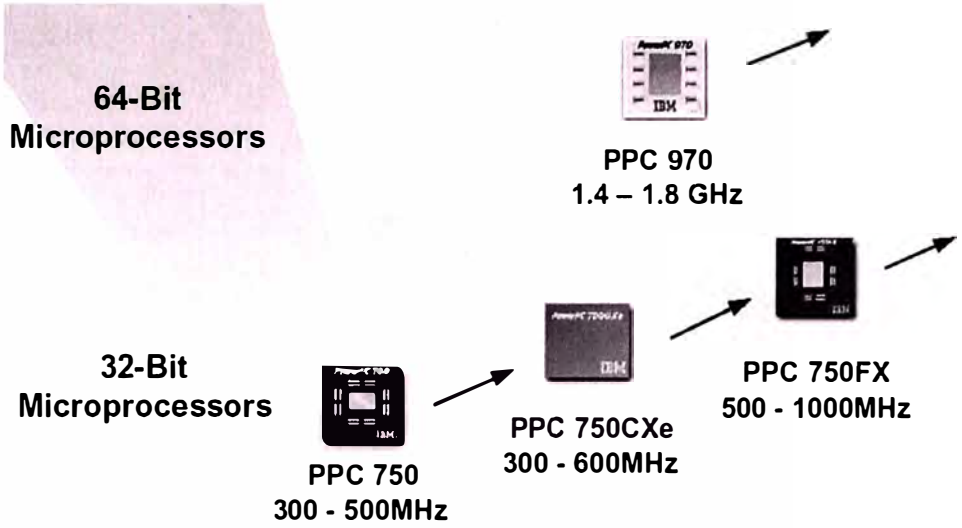


Figura 3.2 Familia de Microprocesadores PowerPC

Para el caso de estudio nuestro, profundizaremos sobre el modelo 9xx el cual es un microprocesador de 64bits y nos permitirá compararlo con otros modelos del mercado.

3.3 Funcionamiento del Procesador PowerPC 970

El PowerPC 970, fue desarrollado para la ejecución de sistemas de 64 bits y que a su vez permitieran soportar las aplicaciones de 32 bits, esto con la idea de proteger las inversiones de los clientes, su arquitectura robusta y su capacidad de tener un bus de transferencia con capacidad de transmitir hasta casi 7Gb/seg lo hacen uno de los procesadores mas rápidos del mercado de acuerdo con las estimaciones de performance SPECInt 2000 y SPEC fp 2000.

El PowerPC 970 ejecuta un Pipeline de variada fase de profundidad, dependiendo del tipo de SIMD (single instruction multi data). Para la etapa de **Fetch** y **Decode** puede ejecutar hasta 8 instrucciones por ciclo, para la etapa de **Dispatch** hasta 5 instrucciones por ciclo para las unidades de función, en la etapa de Ejecución hasta 8 instrucciones por ciclo en las unidades de función, la entrega de resultados realiza hasta 5 instrucciones por ciclo.

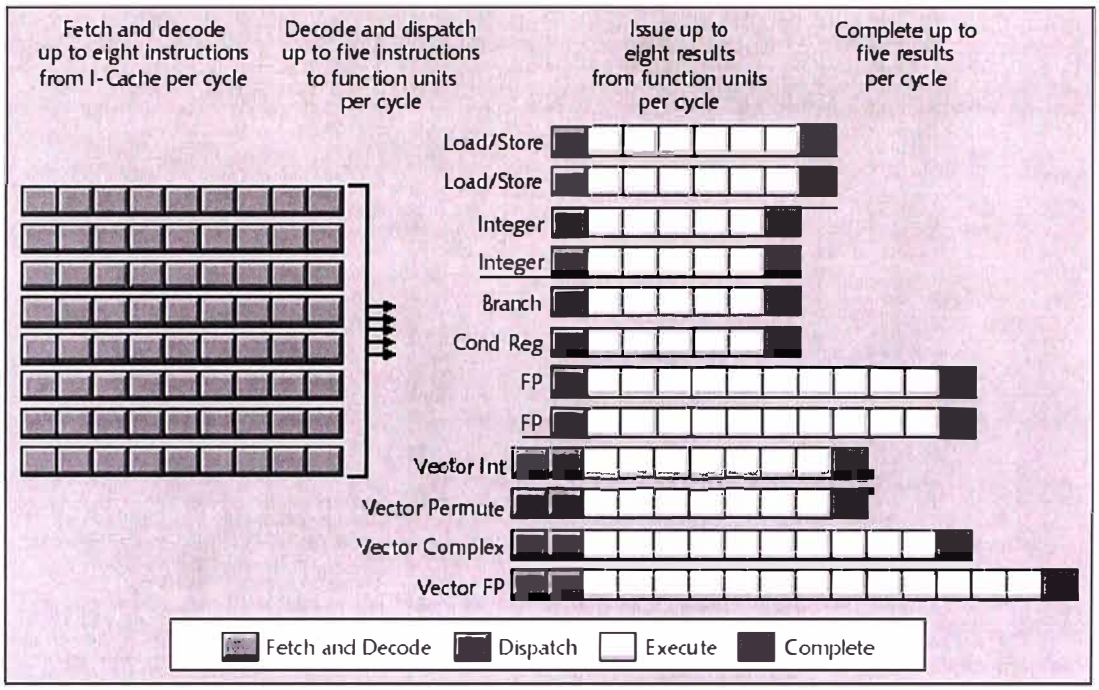


Figura 3.3 Diagrama de instrucciones del PowerPC 970

La arquitectura interna permite obtener esta alta capacidad de procesamiento, gracias al Altivec extensión (VMX), se pueden ejecutar hasta 12 funciones de proceso, con su ingeniería de vector el cual contiene 128bits de ancho de data, posee 2 sub-unidades de función para enteros y punto flotante(2 vectores de ALU para enteros, 1 vector de FPU y 1 vector de permutación), 32 vectores de registros (data) y 48 vectores renombrados para almacenar operaciones pendientes.

Esta capacidad adicional de vectores hacen posible que en algún momento puedan estar conteniendo hasta 200 instrucciones, de las cuales 100 estarían en la etapa de Dispatch en 20 grupos cada una con 5 instrucciones por grupo, las otras 100 instrucciones estarían en la etapa del Fetch, Decode y Store.

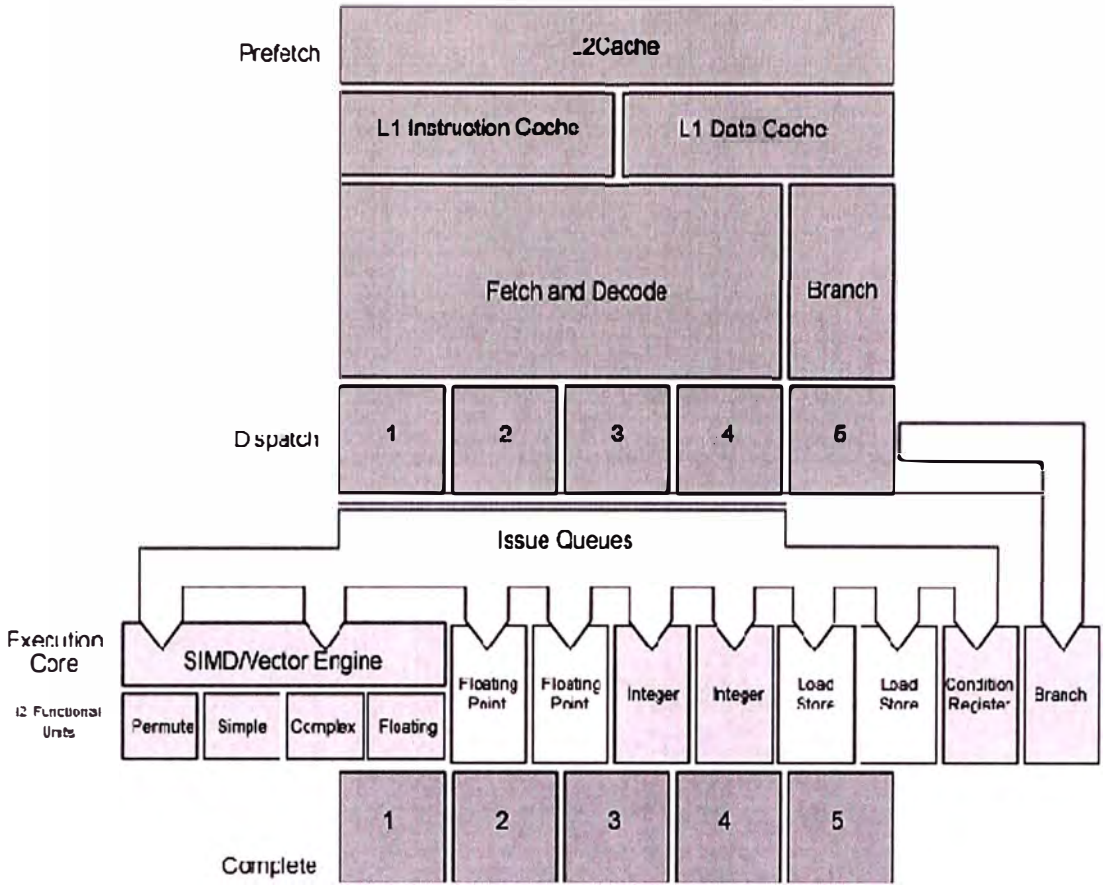


Figura 3.4 PowerPC 970 Arquitectura de Operación interna

3.4 Principales características de la Arquitectura

Esta nueva arquitectura interna del PowerPC incluye un direccionamiento de 64 bits **(7)** de memoria efectivos, y un rango direccionamiento real de 42 bits. Todos los enteros, puntos flotantes, registros son de 64 bits, así mismo tiene 32 registros de enteros para propósito general y 32 registros de punto flotante, también se incluyen 48 registros de 64bits para enteros y 48 registros para punto flotante que son utilizados para almacenar instrucciones pendiente.

La capacidad de computo viene dada por un reloj de 1.8Ghz, integrada en el procesador el cual es particionado hasta en 4 veces su velocidad para sincronizar la ejecución en los buses de datos (lectura y escritura) a memoria, procesando a una frecuencia de 450Mhz.

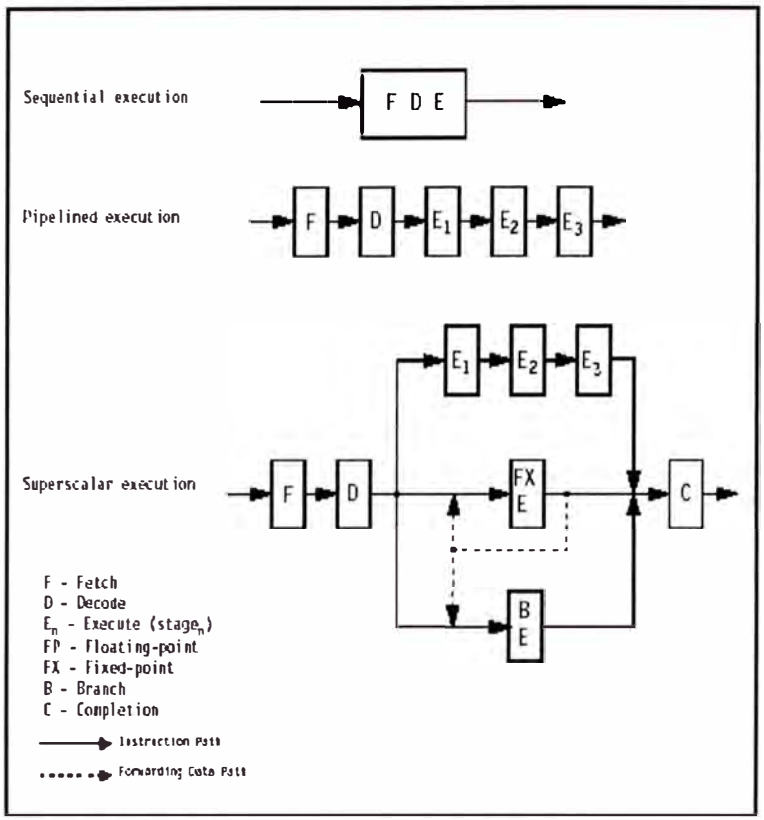


Figura 3.5 Proceso del Pipeline en el PowerPC 970

La memoria cache de nivel L2 integrada es de 512Kbytes, posee 2 buses uni-direccionales de 32Kbits para lectura y escritura desde la memoria real, dicha memoria tiene hasta 8 caminos de acceso, así mismo la memoria de nivel L1 tiene 2 componentes una para las instrucciones de 64Kbyte y otra para los datos de 32Kbytes ambas se conectan al nivel L2 mediante dos buses Bi-direccionales de 256Kbits cada una.

3.4.1 Arquitectura PowerPC 970

En la figura 3.5 podemos observar que las componentes de procesamiento son 12 funciones bien definidas de acuerdo al tipo de información a procesar, esta nueva arquitectura se beneficia de 2 unidades de proceso Lógico Aritmético (ALUs), 2 unidades de punto flotantes (FPUs), 2 unidades de carga y almacenamiento (Load/Store Unit), 1 unidad de condición de registro, 1 unidad de branch, y 4 vectores para procesar enteros simples, enteros complejos, punto flotante y una para permutación. Con todas estas funciones se puede procesar más rápidamente aplicaciones que requieren computo intenso.

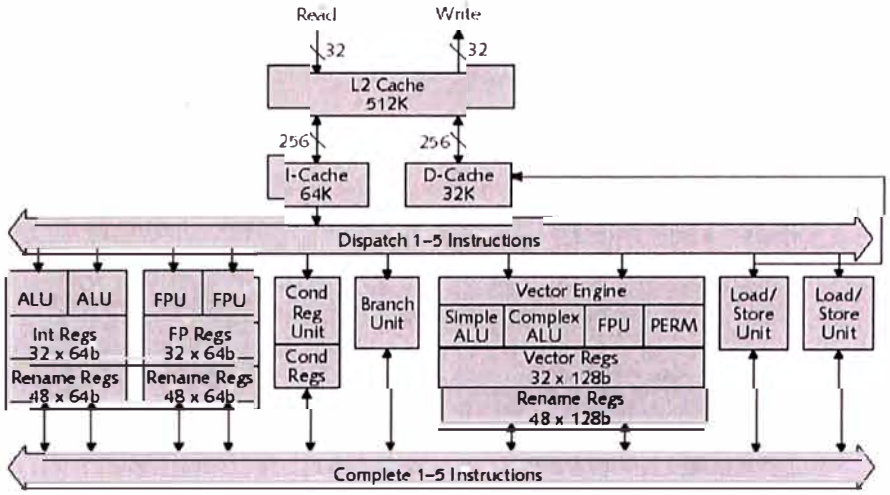


Figura 3.6 Arquitectura del PowerPC 970

3.4.2 Sistema de memoria Cache

El sistema de memoria cache esta conformado por los buses de conexión al CPU, al bus de acceso a dispositivos externos, por la memoria cache de nivel L2 y la memoria no cache , así mismo tiene un controlador de memoria. En la figura 3.6 podemos apreciar este arreglo.

- La unidad de interfase al core (CIU) contiene una instrucción de Fetch de puerto (IFU), dos instrucciones de carga y almacenamiento al puerto (LSU), y una instrucción de Fetch de datos con traslado al puerto. El CIU ejecuta el ordenamiento de arbitraje, las colas y la carga de información en el puerto, el canal de data es 16 byte hacia el CPU.

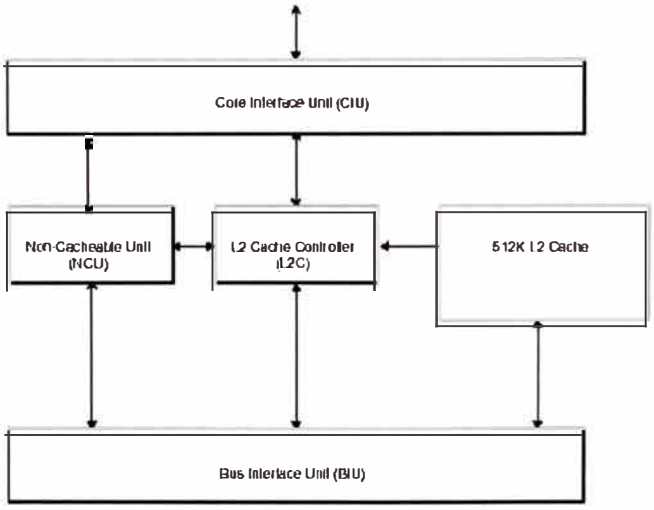


Figura 3.7 Diagrama de bloques del sistema de memoria cache

- El controlador de cache L2 (L2C) reside entre el (CIU) y el (BIU) también tiene conexiones con NCU (unidad no cacheable), la frecuencia de funcionamiento es la misma que la del CPU, permite la carga y almacenamiento de toda la información, envía información de 8 palabras para la carga de data, envía registros de 8 palabras para las instrucciones Fetch, maneja una cola de 6 lecturas y 8 (64byte) de almacenamiento, soporta y gatilla el almacenamiento, recupera información errada desde su directorio de redundancia.
- La unidad no cacheable (NCU) permite ayudar a todas las informaciones que requiere el CPU y que no pasan por el cache L2. Las instrucciones de operación de cache y sincronismo son enviadas al NCU, este almacena dichas instrucciones mientras son requeridas, la diferencia con la unidad de cache L2 es que esta unidad no maneja data asociada.
- La unidad de interfase al Bus (BIU) permite conectar los dispositivos de entrada y salida con el PowerPC, el bus tiene una alta frecuencia de conexión punto a punto, porque su estructura solo permite manejar un segmento a la vez. El BIU contiene la lógica requerida para el arbitraje en medio del constante requerimiento de diversos dispositivos, el BIU también contiene lógica para respuesta a decodificación de transacciones.

3.4.3 Tecnología de Vector (VMX)

La tecnología de vector expande la arquitectura del PowerPC a través de la adición de un vector de 128bits como unidad de ejecución, el cual opera con las existentes unidades de enteros y punto flotante. Esta nueva tecnología ofrece una alta capacidad de operaciones en paralelo, permitiendo una ejecución simultanea de hasta 4 operaciones de 32bits de punto flotante o 16 operaciones de 8bits en una sola instrucción. Todos los vectores son de 128bits y trabajan como pipeline, están pensadas para trabajar de forma análoga a las unidades de punto flotante.

Las unidades de punto flotante fueron previstas para soportar una alta precisión en cálculos científicos y la tecnología de vector son adicionadas para acelerar el siguiente nivel de manejo de performance sobre grande anchos de banda en comunicaciones y aplicaciones de computo.

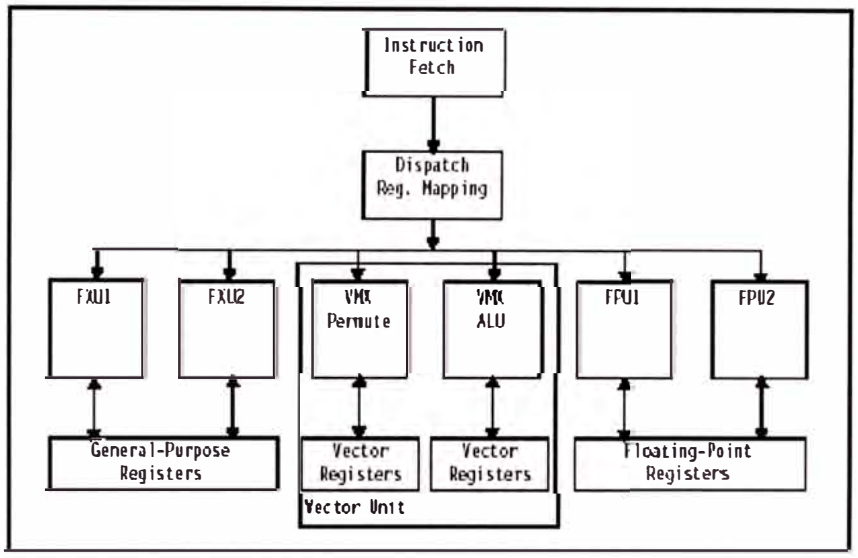


Figura 3.8 Estructura del vector VMX

La Unidad de proceso del vector tiene 2 unidades de envío: un vector ALU y un vector de permutación. El vector ALU esta a su vez divido en un vector de punto flotante (32 bits), un vector de punto fijo simple y un vector punto fijo compuesto. El vector ALU y el de permutación reciben instrucciones precodificadas vía la cola de la unidad de secuencia de instrucciones. Los vectores de carga, almacenamiento y envío de instrucciones son ejecutadas en la unidad de carga y almacenamiento (LSU). La tecnología de vector es escalado solo por las piezas necesarias para facilitar la eficiencia en el ciclo de tiempo, la latencia, y la velocidad de transmisión en implementaciones de hardware.

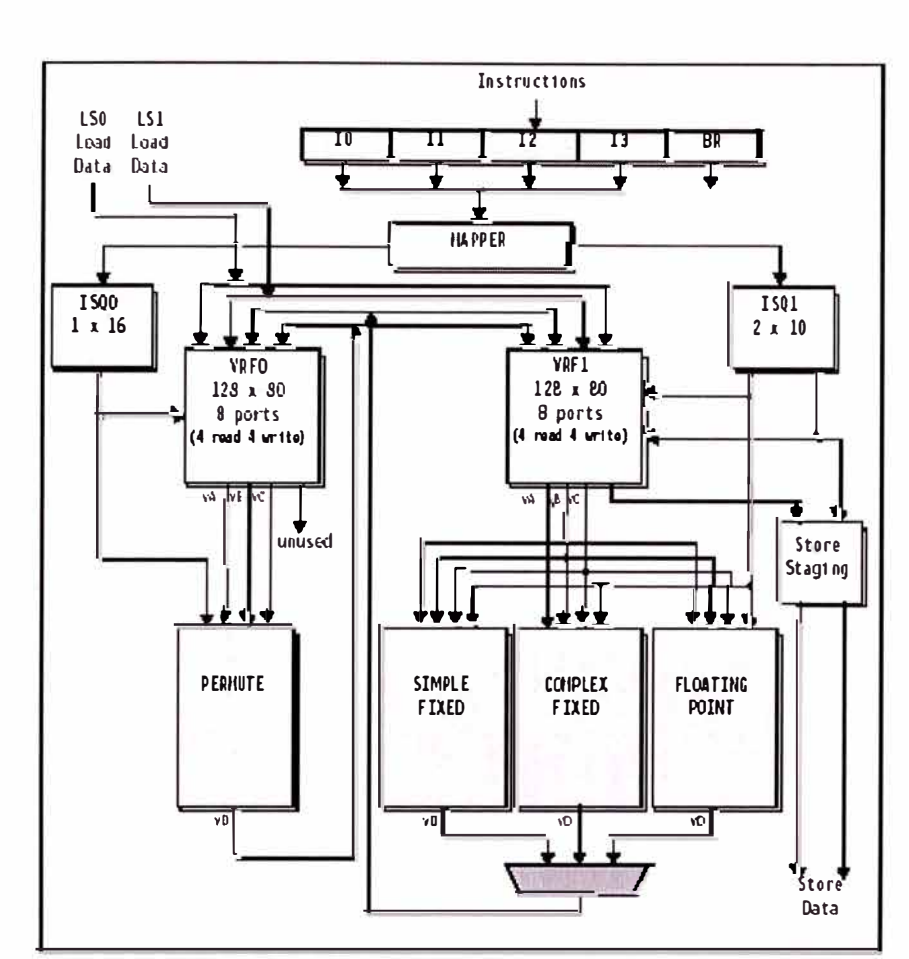


Figura 3.9 Diagrama de funcionamiento de la tecnología de vector

En la figura 3.8 se muestra con mas detalle el diagrama de funcionamiento de la unidad de vectores. Se puede apreciar dos salidas de colas (ISQ0 y ISQ1), la primera puede encolar hasta 16 tipos de instrucciones para ejecución en el vector de permutación, la segunda puede encolar hasta 20 vectores de instrucción en 2 colas para proveer la salida de 2 instrucciones en un mismo tiempo para alguna de las 3 pipeline de ejecución (simple, compleja y de punto flotante). Nota que allí existen 80 vectores de registro, de esos solo 32 registros pueden ser accesados vía las instrucciones, el resto son usados para registros renombrados que almacenan instrucciones, los cuales pueden ser requeridos en cualquier momento para alguna ejecución.

3.4.4 Características adicionales

En resumen las ventajas son las siguientes:

- Ha aprovechado las ventajas del procesador POWER4 de 64 bits
- Representa una nueva familia de procesadores PowerPC

- Proporciona alta performance para procesamiento de propósito general
- Tiene un diseño superescalar con múltiples unidades para ejecución "pipeline"
- Presenta mejoras para multimedia, gráficos y movimiento de datos a través de una implementación de hardware de una facilidad de procesamiento SIMD (single instruction multiple data - una instrucción múltiples datos)
- Soporta las demandas de ancho de banda de un diseño altamente superescalar y la facilidad SIMD extendida a través de un centro de alta velocidad en el bus del procesador
- Fabricado con tecnología IBM para procesos de 0.13 manómetros, con menor consumo de energía y pequeño tamaño.
- .Su diseño a 64 bits soporta doble punto flotante y doble punto fijo y doble unidad carga / almacenamiento de datos a 64 bits, por medio de unidades SIMD de 128 bits de data paths.
- Tiene 512 Kb de cache L2 interno
- Obtiene altas performance de los sistemas con hasta 1 GHz de bus del procesador, con paths dedicados a la lectura y escritura para eliminar los cuellos de botella de performance.

3.4.5 Especificaciones Técnicas del PowerPC 970

La siguiente tabla muestra sus características técnicas más saltantes:

Tabla 3.1 Diseño de Implementación del PowerPC 970

Target Frequencies	1.4 to 1.8 GHz
Architecture	64-bit PowerPC, 32-bit compatible
Performance*	937 SPECint2000 @ 1.8 GHz 1051 SPECfp2000 @ 1.8 GHz 5220 DMIPS @ 1.8 GHz (2.9 DMIPS/MHz)
Caches	64KB, I cache, w/parity 32KB, D cache, w/parity 512KB, L2 cache, w/ECC
Voltages	1.3V core logic and I/Os
Typical Power Dissipation*	42W @ 1.8 GHz, 1.3v 19W @ 1.2 GHz, 1.1v
Package	25x25mm CBGA 576 pins on 1mm pitch (161 signals)
Technology	0.13µm, CMOS w/ SOI 8 levels of copper interconnect
Target Schedule*	Samples 2Q 2003, Production 2H 2003

3.5 Set de Instrucciones

Las instrucciones **(8)** que el procesador puede ejecutar caen en tres clases:

- Instrucciones de la branch
- Instrucciones de punto fijo
- Instrucciones de punto flotante

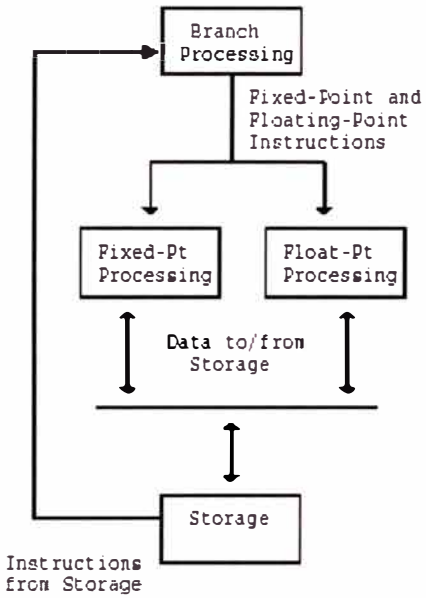


Figura 3.10 Modelo Lógico de Instrucciones

Las instrucciones pueden ser de tres tipos:

- Clase de instrucciones definidas, en general son instrucciones que se garantizan en todas las implementaciones con Procesador PowerPC, la única excepción son las instrucciones opcionales. Una instrucción definida puede tener un formato invalido de and/or.
- Clase de instrucciones ilegales, son instrucciones definidas para versiones futuras de Implementación en Procesadores PowerPC, una acción de ejecutar cualquier de estas instrucciones ocasionara un error en el sistema.}

- Clase de instrucciones reservadas, son instrucciones permitidas para propósitos específicos, la ejecución de alguna de estas instrucciones puede causar mejoras en la performance o un error en el sistema.

Para hacer uso de las instrucciones se requieren un set de registros, tales como:
Condición de Registro-CR (32 bits)
Registro de enlace-LR (64 bits)
Registro contador-CTR (64 bits)
Registro de Propósito General-GPR (32 registros de 64bits c/u)
Registro de excepción de punto fijo-XER (64 bits)
Registro de punto flotante-FPR (32 registros de 64bits c/u)
Registro de control y estatus de punto flotante-FPSCR (32 bits)
Así mismo la nomenclatura para los datos es la siguiente: Byte (8 bits), Halfword (2 Bytes), Word (4 Bytes), Doubleword (8 Bytes), Quadword (16Bytes).

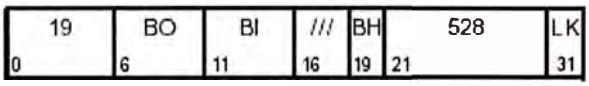
3.5.1 Instrucciones de la Branch, computa el efectivo direccionamiento (EA) objetivo de los siguientes cuatro medios:

- Adicionar un desplazamiento de dirección de la instrucción de la branch (Branch Conditional con AA=0).
- Especificando una dirección absoluta (Branch Conditional con AA=1)
- Usando la dirección del Registro de enlace (LR)
- Usando la dirección del Contador de Registro (CTR)

Ejemplo con el uso de dirección del Contador de Registro (CTR)

Branch Conditional to Count Register
XL-form

bcctr BO,BI,BH (LK=0)
bcctrl BO,BI,BH (LK=1)
|POWER mnemonics: bcc, bccl|




```
cond_ok ← BO0 | (CRBI = BO1)
if cond_ok then NIA ←64 CTR0:61 || 0b00
if LK then LR ←64 CIA + 4
```

El campo BI especifica la condición del bit de Registro a ser testeado. El campo BO es usado para resolver la instrucción de la branch.

Si LK=1 después la dirección efectiva de la instrucción siguiente de la instrucción branch es colocada en el Registro de enlace.

Si el decremento y test del CTR opcional es especificado, la instrucción forma es invalida.

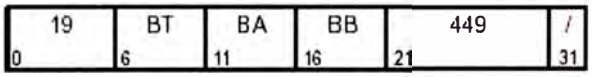
La misma operación en Mnemónico para la operación de contador de registro es la siguiente:

Extended:	*Equivalent to:*
bcctr 4,6	bcctr 4,6,0
bltctr	bcctr 12,0,0
bnectr cr2	bcctr 4,10,0

Otro ejemplo utilizando Instrucciones lógicas de condición de registro

Condition Register OR XL-form

cror BT,BA,BB



$CR_{BT} \leftarrow CR_{BA} \mid CR_{BB}$

El bit en la condición de registro especificada por BA es ORed con el bit en la condición de registro especificad por BB, y el resultado es colocado en el bit de la condición de registro especificada por BT.

Registro Especial Alterado CR ^{BT}

Mnemónico extendido:

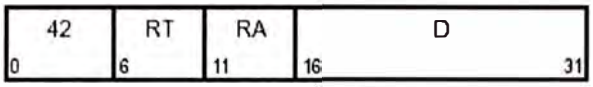
Extended:	*Equivalent to:*
crmove Bx,By	cror Bx,By,By

3.5.2 Instrucciones de punto fijo, todas las manipulaciones están hechas sobre los registros de propósito general (GPR)y sobre el registro excepción de punto fijo (XER). Son las instrucciones de carga, acceso, almacenamiento, suma, sustracción, multiplicación, división y lógica and, or y xor.

Algunos ejemplos de uso de dichas instrucciones:

Load Halfword Algebraic D-form

lha RT,D(RA)



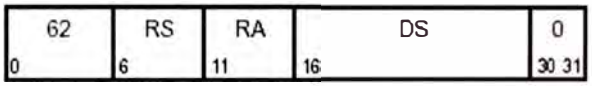

```
if RA = 0 then b ← 0
else          b ← (RA)
EA ← b + EXTS(D)
RT ← EXTS(MEM(EA, 2))
```

Permite que la dirección efectiva (EA) sea la suma de (RA|0)+D. La media palabra en la dirección almacenada por EA es cargada en RT $_{48:63}$.

RT $_{0:47}$ están llenados con una copia de bit 0 de la cargado en la media palabra.

Store Doubleword DS-form

std RS,DS(RA)




```
if RA = 0 then b ← 0
else          b ← (RA)
EA ← b + EXTS(DS || 0b00)
MEM(EA, 8) ← (RS)
```

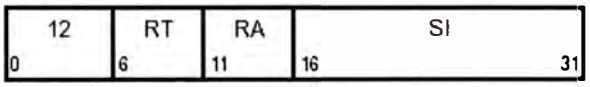
Permite que la dirección efectiva (EA) sea la suma de (RA|0)+(DS||0b00).

(RS) es almacenado en la doble palabra en la dirección almacenada por EA.

Add Immediate Carrying D-form

addic RT,RA,SI

[POWER mnemonic: ai]

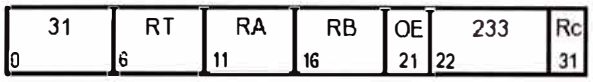



```
RT ← (RA) + EXTS(SI)
```

La suma de RA más SI es colocada en el registro RT.

Multiply Low Doubleword XO-form

mulld	RT,RA,RB	(OE=0 Rc=0)
mulld.	RT,RA,RB	(OE=0 Rc=1)
mulldo	RT,RA,RB	(OE=1 Rc=0)
mulldo.	RT,RA,RB	(OE=1 Rc=1)



$prod_{0:127} \leftarrow (RA) \times (RB)$
$RT \leftarrow prod_{64:127}$

Los operadores de 64bits son RA y RB. El orden bajo de 64bits del producto de 128bits de los operadores son colocados en el registro RT.

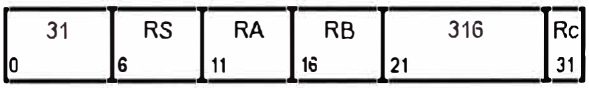
Si OE=1 despues OV es puesto en 1 si el producto no puede ser representado en 64 bits.

Ambos operadores y el producto son interpretados como signos enteros.

Registros alterados: CR0 (si Rc=1), SO OV (si OE=1).

XOR X-form

xor	RA,RS,RB	(Rc=0)
xor.	RA,RS,RB	(Rc=1)



$RA \leftarrow (RS) \oplus (RB)$

El contenido del registro RS son XORed con el contenido del registro RB y el resultado es colocado en el registro RA.

Registro especial alterado: CR0 (si Rc=1).

3.5.3 Instrucciones de punto flotante, son instrucciones que proveen mejorar la performance en cálculos aritméticos, de conversión, de comparación, y otras operaciones en registro de punto flotante, también para mover data de punto flotante entre la información almacenada y los registros, y para manipular el estado y control de los registros. Estas instrucciones se dividen en 2 categorías:

- Instrucciones Computacionales, son todas las instrucciones de suma, sustracción, multiplicación, división, extracción de la raíz cuadrada, conversión, comparación y la combinación de todas ellas. Estas instrucciones proveen operaciones de punto flotante y colocan un estatus de información en el registro de control y estatus de punto flotante (FPSCR).
- Instrucciones No computacionales, son todas las instrucciones de carga, almacenamiento, movimiento de contenidos en registros de punto flotante, posibilidad de alterar los signos, manipuleo de estatus del registro de control y estatus, etc. Todas estas instrucciones no generan cambios en el (FPSCR) a excepción de la instrucción de manipuleo.

Tabla 3.2 IEEE Campos de Punto Flotante

	Format	
	Single	Double
Exponent Bias	+127	+1023
Maximum Exponent	+127	+1023
Minimum Exponent	– 126	– 1022
Widths (bits)		
Format	32	64
Sign	1	1
Exponent	8	11
Fraction	23	52
Significand	24	53

Representación de los valores en punto flotante, la maquina representa tres valores de aproximación de números reales: números normalizados, números desnormalizados y valores de cero.

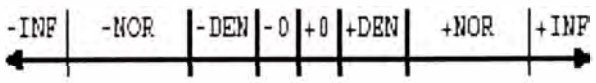
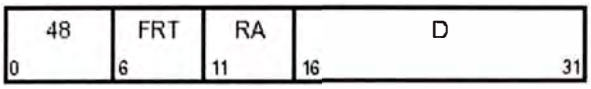


Figura 3.11 Aproximación de los números reales

Ejemplos con instrucciones de punto flotante:

Load Floating-Point Single D-form

lfs FRT,D(RA)



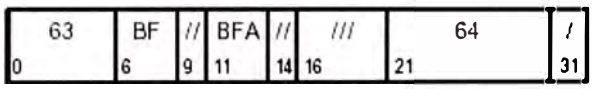

```
if RA = 0 then b ← 0
else          b ← (RA)
EA ← b + EXTS(D)
FRT ← DOUBLE(MEM(EA, 4))
```

Permite que la dirección efectiva EA sea la suma de (RA|0)+D.

La palabra almacenada por la dirección EA es interpretada como punto flotante de simple precisión. Esta palabra es convertida a formato doble de punto flotante y colocada en el registro FRT. No se altera ningún registro.

Move to Condition Register from FPSCR X-form

mcrfs BF,BFA



El contenido de FPSCR campo BFA son copiados al campo de registro de condición BF. Todas los bits copiados son puestos cero en el FPSCR. Si el bit FX es copiado, este es puesto en cero en el FPSCR.

Los registros alterados son los siguientes:

```
CR field BF
FX  OX                          (if BFA=0)
UX  ZX  XX  VXSNAN              (if BFA=1)
VXISI  VXIDI  VXZDZ  VXIMZ      (if BFA=2)
VXVC                            (if BFA=3)
VXSOFT VXSQRT VXCVI             (if BFA=5)
```

3.6 Costo aproximado de un Servidor con tecnología PowerPC 970

IBM ha llamado a sus servidores con tecnología PowerPC **(9)**, eServer BladeCenter JS20. Una configuración con 2 procesadores PowerPC 970 de 2.2Ghz, con 70 GB en discos de 10,000 rvp y memoria cache 4Gbyte, esta costando alrededor de US$10,000.00

CAPITULO IV
HERRAMIENTAS DE MONITOREO

4.1 Conceptos Básicos

Existen dos tipos de herramientas para medir sistemas computacionales bajo una cierta carga:

4.1.1 Monitores de Software

Programas que detectan los estados de un sistema; o conjuntos de instrucciones (sondas de software) que son capaces de detectar eventos.

4.1.2 Monitores de Hardware

Dispositivos electrónicos conectados en puntos específicos del sistema, en donde detectan las señales (niveles de voltaje o pulsos) que caracterizan el fenómeno en observación.

Como toda herramienta de medición de fenómenos físicos, los instrumentos de monitoreo para computadores consumen alguna energía del sistema que se mide. Obviamente, este problema atañe más a los monitores de software que a los de hardware. Por ejemplo, los monitores de software utilizan espacio de memoria, tiempo de CPU y ejecutan operaciones de E/S.

A pesar de que los monitores de software incrementan en cierta medida la carga en el sistema, ellos tienen muchas otras ventajas respecto a los monitores de hardware.

Otra característica de las herramientas de medición es su precisión, la cual puede ser expresada por el nivel de error en los valores obtenidos. En el caso de monitores de software de muestreo, el usuario puede evaluar, e incluso influenciar, la precisión de las mediciones. Para otras herramientas sólo queda confiar en las especificaciones del fabricante.

En la figura 4.1 se muestra un esquema conceptual de una herramienta de medición. La interfaz de **instrumentación** en el sistema en observación incluye, por ejemplo: instrucciones insertadas en el sistema operativo, cronómetros, pins de conexión, etc.

El elemento **selector** permite filtrar y observar en forma "selectiva" las actividades medibles. El elemento **procesador** efectúa pruebas y cálculos con la información del estado de los componentes del sistema en observación. El elemento **almacenador** guarda (en disco y/o cinta) la información proporcionada por el elemento procesador, para que en una etapa posterior sea analizada y reportada por el elemento interpretador.

Un **monitor en tiempo real** efectúa la fase de interpretación en paralelo con las otras fases del proceso de medición.

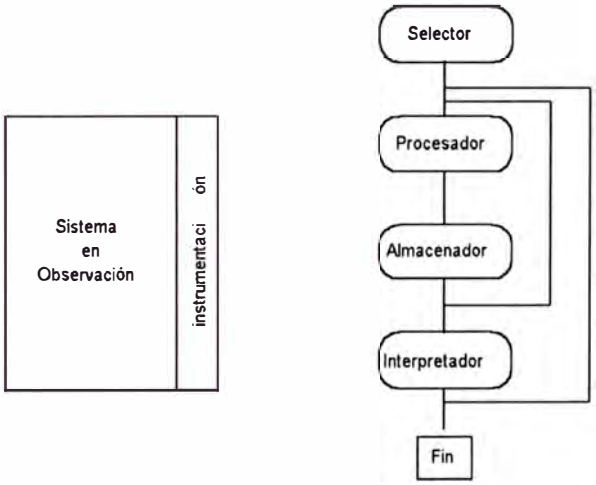


Figura 4.1 Esquema Conceptual de una Herramienta de Medición

4.2 Tipos de Monitores de Software

Existen dos tipos básicos de monitores de software:

- Monitores detectores de eventos ("event-driven").
- Monitores de muestreo ("samplers").

Existen herramientas que combinan elementos de detección de eventos y de muestreo. Los monitores de software pueden utilizar distintas técnicas para coleccionar información:

- Agregar un programa
- Modificar el software a ser medido ("software probes")

- Modificación del sistema operativo o de un programa del sistema

4.2.1 Monitores Detectores de Eventos

Los monitores detectores de eventos generan bastante más "overhead" (consumo no productivo de recursos) que los de muestreo, pero son más precisos.

Un evento es cualquier cambio en el estado de un sistema. Por ejemplo:

- Comienzo o fin de una operación de E/S.
- Transición de la CPU del estado "ocupada" a "ociosa" o vice-versa.
- Detección de un "page fault".
- Inicio y termino de un programa.
- Comienzo de una búsqueda en el disco ("seek").

Evento de software: relacionado con las funciones de programas (por ejemplo, comienzo de una operación de E/S).

Evento de hardware: aparición de señales en los circuitos de un componente del sistema.

Muchos eventos de hardware pueden ser reconocidos a través de software.

Típicamente, para detectar un evento de software se inserta un código especial ("hook") en lugares específicos del sistema operativo.

Cuando el evento en observación ocurre, el código especial hace que el control se transfiera a una rutina apropiada. La rutina registra la ocurrencia del evento y almacena datos relevantes en un área de "buffer". Luego se retorna el control al sistema operativo.

La información recolectada por este tipo de monitor se conoce como **rastro de eventos** ("event trace").

Reducción de datos: el conjunto de información recolectada (almacenada en disco/cinta) es procesada posteriormente por un programa de análisis.

El "overhead" ocasionado por un monitor de eventos es **directamente proporcional** al número de eventos interceptados y sus frecuencias.

El uso de **"buffers"** en memoria principal puede ocasionar más overhead (retardo) o hacer que se pierda información.

En la mayoría de los casos, la detección de eventos por software es difícil de construir y ocasiona un "overhead" considerable en el sistema.

4.2.2 Monitores de Muestreo ("Samplers")

Los monitores de muestreo pueden ser de dos tipos:

Batch : la reducción de datos es posterior a la medición

En línea : la reducción de datos es en línea (pantalla)

Con los monitores de muestreo es posible controlar la interferencia ("overhead") que ellos ocasionan en el sistema, por medio de los siguientes parámetros:

- Selectividad de las mediciones
- Variabilidad de los intervalos de muestreo (frecuencia)

Por ejemplo:

Sea un procesador de 1 MIPS (millón de instrucciones por segundo) y un monitor de muestreo genera 4.000 instrucciones/medición

Entonces puede detectar una considerable cantidad de datos.

Los monitores de muestreo utilizan un **cronómetro** para interrumpir la actividad del sistema a intervalos regulares de tiempo para la extracción de datos.

Intervalos de muestreo aleatorios serían más apropiados pero típicamente se utilizan **intervalos largos uniformemente**, pues son más fáciles de construir.

La información recolectada se agrupa en **registros lógicos**, de acuerdo a intervalos de tiempo escogidos por el usuario (típicamente 2 a 10 minutos). El registro lógico es el **conjunto más pequeño de datos** en el cual el analizador calcula valores medios, porcentajes y otras estadísticas.

4.2.3 Mediciones de Tiempo

Un monitor de software recoge datos de los estados y actividades de componentes del sistema en un número de intervalos de tiempo.

De esta forma, se puede determinar la duración de los fenómenos de interés, como por ejemplo:

- Utilización de la CPU
- Frecuencia alta de paginación
- Utilización de periféricos
- Tempo promedio entre el inicio de una solicitud de página y su llegada a memoria

Por otro lado, si existe un reloj del sistema que puede ser accesado desde programas, entonces se pueden efectuar **mediciones directas** de tiempo.

Se pueden obtener **mediciones directas** de la diferencia entre el tiempo en que el fenómeno terminó y el tiempo en que comenzó. Desgraciadamente este método no siempre puede ser utilizado debido a interrupciones temporales y suspensión de la actividad observada debido a otras operaciones de mayor prioridad.

Aunque se pueden sumar los tiempos en cada período de actividad, este método es poco preciso y es difícil de implementar.

Típicamente, un sistema computacional posee un **reloj físico** el cuál es utilizado por el sistema operativo para la asignación de CPU y otros recursos. El sistema mantiene también un **reloj lógico** (o virtual) para cada proceso de modo de controlar la asignación de CPU.

El **reloj lógico** puede ser utilizado para monitorear el consumo de recursos por proceso, pero puede existir un retardo entre el comienzo/fin de una actividad y el uso del reloj lógico asociado y el sistema operativo puede cargar a un proceso el tiempo de manejo de ciertas interrupciones que no corresponden a ese proceso.

4.3 Monitores de Contabilidad

El monitor o sistema de contabilidad ("accounting") es una parte integral de la mayoría de los sistemas multiusuarios. Aunque su objetivo principal es permitir el cobro del uso del sistema computacional a los usuarios, es de gran utilidad para la medición del rendimiento.

Para todos los efectos prácticos, funciona en forma similar a un monitor de software, generalmente por detección de eventos. En este monitor se incluye la información de los procesos que son ejecutados bajo el sistema:

- Tiempo de CPU utilizado.
- Número de operaciones de E/S a los diferentes dispositivos.
- Utilización del canal de E/S.
- Utilización de la memoria.
- Tiempo de conexión para usuarios interactivos.
- Tiempo de comienzo y fin para cada proceso.

Los monitores de contabilidad en general no proveen:

- Detalle del uso de los diferentes módulos del sistema operativo.
- Detalles del uso de recursos de cada una de las transacciones interactivos o de administradores de bases de datos.
- Tiempo que los usuarios interactivos conectados estuvieron inactivos.

Un sistema de contabilidad tiene muchas características de un monitor de eventos. Sus datos son acumulados en la tabla de procesos del sistema y cuando un proceso termina, esta información es registrada en un archivo antes que su lugar sea ocupado por un nuevo proceso.

4.3.1 Monitoreo de Procesos

Una función primordial de los monitores de contabilidad es el registro de las actividades y consumos de recursos que efectúan los procesos que se ejecutan en el sistema.

Los sistemas multiusuarios se caracterizan por mantener activos múltiples procesos de diversos usuarios. Para esto es necesario que el procesador cambie su atención de un proceso a otro según algún mecanismo (por ejemplo tiempo compartido o "time sharing") para atender a todos los procesos. Cada proceso tiene un ambiente en el cual se desenvuelve. Este ambiente es diferente para cada proceso en el sistema. La forma de almacenar estos diferentes ambientes es mediante una tabla en la cual están registrados todos los procesos activos en el sistema (esta es la representación mas usual de encontrar).

Aprovechando la existencia de esta tabla con el ambiente para cada proceso, es usual que se utilice para registrar también el uso que hacen de los recursos. Por ejemplo el tiempo de uso acumulado de CPU y el número de operaciones de E/S, entre otros. Cuando un proceso termina, el registro del proceso en la tabla es eliminado vaciando su contenido en un archivo. Posteriormente, otros programas son usados para reducir estos datos convirtiéndolos en información de gastos contables.

La forma básica de funcionamiento de un monitor de contabilidad es la siguiente:

- Se genera una entrada en la tabla de procesos para cada proceso que comienza.
- Se inicializa el ambiente del proceso y de los contadores de actividad.

- Se registran los cambios en los contadores cada vez que el proceso es afectado por ellos.
- Se vacía el contenido de los contadores en un archivo y se elimina la entrada en la tabla.

La administración de los archivos de resultados de los procesos es variada y depende del sistema. Puede ser un archivo único o archivos múltiples dependiendo del tamaño de los mismos. Puede producirse una reducción de datos, es decir se acumulan los procesos idénticos cada vez que el sistema trabaja con archivos muy grandes.

El tamaño del archivo de contabilidad es un factor que debe tomarse en cuenta en los análisis, puesto que puede llegar a tener varios megabytes por día de operación. Si se conservan todos estos datos, rápidamente pueden consumir el espacio en disco alterando el normal desempeño del sistema.

4.3.2 Limitaciones de los Monitores de Contabilidad

El monitoreo de contabilidad tiene un problema serio. Este es que la información de los procesos no puede ser registrada hasta que los procesos hayan terminado. En los sistemas computacionales existen procesos que nunca terminan por lo tanto la actividad de ellos nunca será registrada por este tipo de monitor.

Otro problema relacionado aparece al introducir el intervalo de muestreo. Para ilustrar este problema veamos el siguiente ejemplo (Figura 4.2).

Caso 1: El proceso A parte en **t=-10** horas, se inicia el monitoreo en **t=0**, en **t=1** hora termina el proceso A y en **t=2** horas termina el monitoreo. El tiempo de muestreo fueron 2 horas, pero el tiempo de ejecución del proceso A es de 11 horas, lo que a primera vista parece una contradicción.

Caso 2: El monitoreo parte en t=0 y el proceso A parte en **t=1** hora, el monitoreo termina en t=2 horas y el proceso A en t=3 horas. En este caso, el proceso A no quedó registrado. Por lo tanto, aparentemente no se pueden establecer los recursos ocupados por él.

Consideremos en este mismo caso al proceso B que comenzó justo en t=0 y terminó en t=**2**. Supongamos que tuvo un consumo acumulado de CPU de 1 hora. Se podría entonces llegar a la conclusión errónea que el dispositivo CPU estuvo con una utilización del 50% (al considerar sólo el consumo de B, pues el de A no quedó registrado).

Pero si el proceso A también hubiera tenido un consumo acumulado de CPU de 1 hora, la CPU habría estado en realidad ocupada al 50% entre **t=0** y 1 y luego al 100% entre **t=1** y **t=2**, obteniéndose un promedio de 75% de uso de la CPU entre **t=0** y **t=2.**

Caso 3: Si consideramos una combinación de los dos casos anteriores, se puede dar la situación en que el proceso A comienza antes del comienzo del monitoreo (**t=0**) y termina después, produciéndose los mismos problemas del caso 2.

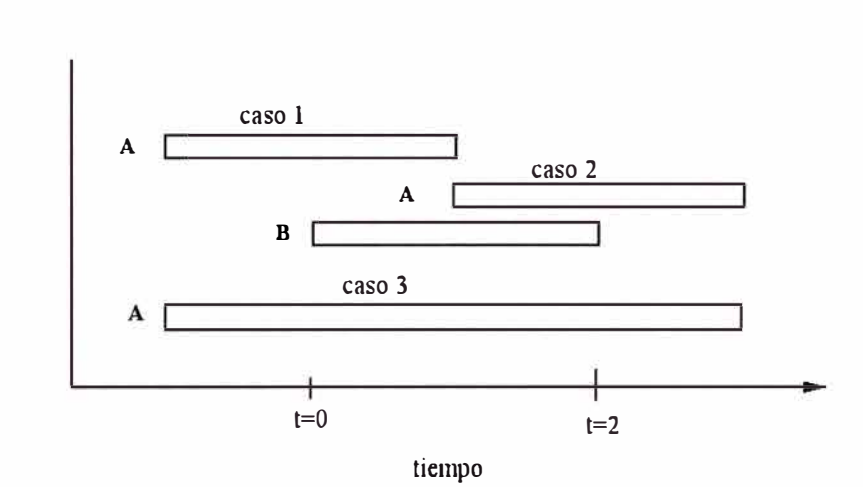


Figura 4.2 Limitaciones de los sistemas de contabilidad

Si bien quedan fuera del alcance de los monitores de contabilidad, todos estos casos son posibles de capturar por medio de monitores especializados. Estos monitores hacen un acceso directo a la tabla de procesos para inicializar el monitoreo, teniendo en cuenta los procesos que están activos y al finalizar el intervalo de observación, registran explícitamente los procesos que aún siguen funcionando. Este acceso a la tabla de procesos no es fácil y también es lento pues la prioridad de un proceso para tener acceso a la tabla es menor que la del Kernel del sistema para modificarla. Prácticamente, es necesaria una modificación en el sistema operativo para obtener esta funcionalidad. **(10)**

Un segundo problema es la naturaleza incompleta de la información que los monitores de contabilidad entregan para la evaluación del rendimiento. Por ejemplo, usualmente registran el número de operaciones de Lectura/Escritura (E/S) que el proceso solicita, pero no especifica el número real de accesos a disco que estas significan. De hecho, muchas de las operaciones de E/S generadas por un proceso, son capturadas por los "buffers" en memoria principal, no siendo necesario llegar hasta los discos para obtener los datos.

Tampoco se pueden obtener, por medio de los monitores de contabilidad, los accesos a disco diferenciados por cada uno de los dispositivos. Es decir, para un cierto período de observación, se registra el total de operaciones de lectura/escritura realizadas en el subsistema de E/S, pero no el número de operaciones realizada en cada uno de los discos.

Por último, es difícil obtener de los monitores de contabilidad una cuantificación y desagregación del trabajo del sistema operativo realizado para atender a los procesos que han sido ejecutados. Por ejemplo, la paginación y el intercambio ("swapping") que implicó la operación de los procesos.

Los monitores de sistema son monitores de software que capturan la información de consumos y rendimiento del sistema computacional (incluyendo todos los dispositivos), sin hacer distinción entre los distintos procesos que fueron ejecutados.

En general constan de dos fases:

- Colección de datos.
- Presentación de la información.

La colección de datos es realizada extrayendo, de las estructuras de datos de la memoria del sistema, información de los contadores de actividad y luego vaciándola en archivos de memoria secundaria para su posterior presentación.

La presentación de la información es dependiente del software en particular. En general tiene los siguientes modos de presentación:

- Interactivo, el cual recoge información en el momento y la presenta al usuario cada vez que ocurre una muestra.
- Histórico, el cual recoge información almacenada en el archivo que se generó en la colección de datos (típicamente un día de trabajo) y la presenta en la pantalla o en un reporte.

Existe una etapa previa a la presentación de los datos, que generalmente está incluida en el software de presentación, para reducir los datos obtenidos del sistema y hacerlos presentables a los usuario en forma inteligible. Por ejemplo, se cambian los valores de los contadores de actividad de CPU a segundos en que la CPU estuvo ocupada.

Otra de las características de los monitores de sistema es la de controlar el período del tiempo en que se presentará la información. Es decir se puede definir el comienzo y fin de los intervalos de presentación y la granularidad de los datos. Por ejemplo presentación de los datos de las 9:00 horas a las 20:00 horas, con una granularidad de cinco minutos.

También se puede seleccionar la información a desplegar de modo que no se requiere ver siempre toda la información. Dentro de esta opción se puede seleccionar también la forma de presentación de los datos, por ejemplo, presentación tabular o gráfica. La presentación gráfica depende de los dispositivos de despliegue disponibles en el sistema.

Existe una variedad de monitores de sistema disponibles para los diversos sistemas computacionales en el mercado. Entre ellos se encuentran.

- Monitor: Para VAX/VMS
- SPM: Para VAX/VMS
- Monitor: Para VAX y DEC /Ultrix
- Sar: Monitor estándar de Unix System V
- Bars: Para serie A MCP/AS de Unisys

4.5 Monitores para Unix System V

El Kernel de Unix mantiene un conjunto de contadores de actividad en una estructura llamada **sysinfo**. En estos contadores se registran los eventos como los "system calls", "context switching", número de páginas libres y asignadas en memoria, etc. Otros contadores mantienen el estado de los recursos del sistema como la utilización de la CPU en sus modos de operación, "user", "system", "wio" y discos.

Es importante destacar que el Kernel no realiza procesamiento alguno de los datos anteriores, como para controlar el sistema para un mejor rendimiento ("auto tunning").

La forma de acceder a los datos almacenados en el sysinfo es por los utilitarios del sistema. La frecuencia mínima de acceso a estos contadores, por procesos de usuarios, es del orden de los segundos. Sobre estos contadores existen un gran número de interfaces las cuales se presentan a continuación.

Existen generalmente tres tipos de interfaces para acceder a los contadores del sistema para medir rendimiento: comandos, llamados al sistema (system calls) y librerías de C. **Las interfaces de comando** son las más fáciles de usar y son utilizadas para operaciones diarias y rutinarias. Las interfaces de **system calls** y la **librería de C** son usadas también pero para realizar análisis particulares.

System calls:

time, obtiene la hora actual.

times, obtiene el tiempo acumulativo de user y system para el proceso e incluye el tiempo esperando.

Librería de C:

clock, tiempo acumulado de user y system desde una ejecución anterior de clock.

monitor, obtiene el número de veces que se ejecuta una función.

system, acceso a comandos shell u otros.

Comandos:

ps process status, despliega el estado de los procesos activos en el sistema (acceso a la tabla de procesos).

sar, actividad general del sistema

sag sar, gráfico

sadp, descriptor de acceso a disco

time, obtiene el tiempo en los modos user, system y real de un comando

timex, igual a time pero con opciones de colección de datos sar

df, cantidad de bloques de disco libres, ocupados y total.

who, usuarios instantáneos del sistema

hinv, inventario del equipo

netstat, actividad de la red de comunicaciones

CAPÍTULO V
HERRAMIENTAS DE MEDICIÓN
(Benchmark)

Un programa de prueba o **benchmark** se define como un programa o conjunto de programas que evalúan las prestaciones de un sistema informático reproduciendo una carga de trabajo genérica en dicho sistema informático. Al proceso de comparar dos o más sistemas mediante la obtención de medidas se le denomina benchmarking.

En general, los benchmark se agrupan en los denominados paquetes benchmark (benchmark suites), que agrupan diferentes programas para medir diferentes aspectos de un sistema informático.

5.1 Pasos para determinar un Benchmark

Para escoger o diseñar un buen paquete benchmark deben de seguirse los siguientes pasos:

- **Determinar los objetivos de uso del benchmark**. Es decir, ver qué sistema se va a medir, y aislar esa parte del sistema del resto para evitar sus efectos. Además, se debe especificar en qué tipo de cargas de trabajo o entorno se va a evaluar: carga científica o entorno de red como servidor, por ejemplo. En la mayor parte de los casos, el uso o diseño de un benchmark se reduce a "demostrar que nuestra máquina es mejor que la de los competidores", aunque este tipo de objetivos es lo que se debe de evitar.

- **Analizar la carga de trabajo de diversos sistemas**, o de un mismo sistema con diversas aplicaciones, para llegar a una carga de trabajo genérica. Por ejemplo, analizar en qué proporción se ejecutan las instrucciones de coma flotante y las de números enteros, o con qué frecuencia y de qué tamaño suelen ser las peticiones en un servidor de ficheros.
- **Escoger los programas** según los objetivos determinados. Cada programa mide diferentes subsistemas, y, por tanto, esta será una razón para elegirlos. Diferentes programas, además, ejercitan de forma diferente el sistema. Por ejemplo, si se quiere medir las prestaciones de un ordenador como servidor Web, se escogerán programas tales como SPECWeb o WebBench (de la Ziff-Davis Benchmark operation); pero si el mismo sistema se quiere usar como servidor de ficheros en una red local, se deberá usar SPEC-SFS. Los resultados en un benchmark no implican que se obtengan los mismos resultados en otro que mida un aspecto similar del mismo sistema: por ejemplo, el que un procesador tenga buenas prestaciones en coma flotante con simple precisión no implica que tenga las mismas prestaciones en doble precisión; diferentes procesadores optimizan en su diseño diferentes tipos de cálculo. El análisis de la etapa anterior tendrá que tenerse en cuenta a la hora de evaluar los diferentes programas, o de seleccionar cuales programas de un benchmark interesan o no.
- **Escoger las métricas** o mediciones que se van a tomar sobre el sistema. Dependiendo de la parte del sistema que se quiera medir, así se usarán diferentes métricas. Estas métricas se escogerán de acuerdo a la caracterización de la Carga, pero son en general de dos tipos: velocidad de una función del sistema informático, y capacidad de la misma.

  Una métrica de tipo velocidad sería el tiempo que tarda en ejecutarse un programa o grupo de programas; una de tipo capacidad el número de usuarios que es capaz de soportar un sistema o el número de instrucciones por ciclo que puede ejecutar un microprocesador. En general, las métricas son función del nivel funcional del sistema o subsistema que se quiere medir.

- **Se deben de tener en cuenta todos los factores que influyan en el rendimiento**. Algunos de esos factores se variarán durante el estudio (por ejemplo, el número de usuarios simulados por el benchmark), y otros permanecerán fijos; en cualquier caso, todos los factores que influyan en el benchmark deberán de ser los mismos para todos los sistemas que se estudien, si se pretende hacer una comparación entre ellos. Por ejemplo, se deberá usar siempre la misma versión del sistema operativo (a no ser que la eficiencia de éste sea una de las cosas que se midan), iguales versiones del compilador, igual cantidad de memoria, de caché e iniciar la medición bajo las mismas condiciones. Habitualmente, en la presentación de los resultados de los benchmark se suele seguir la regla Full Disclosure, o revelación completa, es decir, una indicación pormenorizada de todas las condiciones bajo las que se lleva a cabo el estudio: fechas, versiones, marcas y modelos y opciones de compilación.

  Finalmente, se debe de llevar a cabo algún tipo de análisis. La presentación de muchos datos no permite extraer conclusiones: todas las mediciones se deben de resumir y sintetizar en una figura de mérito (FDM) que indique, básicamente, cuál es el mejor sistema en el objetivo que se ha marcado, como por ejemplo los índices SPECint95 o BYTEmark usados por SPEC-CPU y Byte, respectivamente.

Los paquetes benchmark, y, en general, las cargas de trabajo de prueba deben de reunir una serie de características para cumplir su función adecuadamente:

- **reproductibilidad:** que sea fácil de reproducir para cualquier tipo de sistema, tanto el programa en sí como los resultados a lo largo de diferentes ejecuciones del mismo programa,
- **compacidad:** que contenga poco código, y que no sobrecargue el sistema ni tarde demasiado tiempo en ejecutarse,
- **compatibilidad:** esta es la característica más importante. La carga de prueba debe de ser compatible con todos los sistemas que sean susceptibles de ser medidos, resolución: el benchmark debe de durar lo suficiente como para que sea fácil diferenciar entre dos sistemas con características similares, y

- **escalabilidad**: es decir, que el mismo benchmark sirva para ordenadores con una gama amplia de velocidades, que siga siendo útil a lo largo del tiempo, y que dé resultados significativos, o sea, que sirvan para diferenciar las prestaciones de los sistemas que se evalúan.

5.2 Aplicaciones de los benchmark

Un benchmark, por tanto, se usará en las siguientes ocasiones:

Adquisición de equipos informáticos. En la mayoría de los casos, un equipo informático se va a usar para una gama amplia de tareas, desde llevar a cabo operaciones comerciales hasta ejecutar juegos de red. Por tanto, en tales casos una carga genérica reproducirá con más o menos exactitud la carga que va a ejecutar el sistema. Los resultados del benchmark servirán para justificar la compra de uno u otro equipo informático; en algunos casos incluso los resultados de un benchmark servirán como certificación de las prestaciones de un equipo, como en el caso de TPC, benchmark del Transaction Processing Council que evalúa las prestaciones de los ordenadores como sistema de proceso de transacciones.

Sintonización de un sistema informático. El ejecutar benchmarks periódicamente sobre un sistema que se está usando, permite ver como se deteriora o, en general, cambia su capacidad a lo largo del tiempo. Además, los benchmarks permiten hallar qué partes del sistema se deben cambiar o mejorar, o cómo ha impactado en el sistema el cambio de alguna de sus partes, o la actualización del sistema operativo o de alguno de sus programas.

Planificación de la capacidad de un sistema informático. De la misma forma que se ha indicado en el punto anterior, la evolución de la carga de un sistema y de los resultados de los benchmarks puede permitir prever qué cambios va a hacer falta llevar a cabo en el futuro, y en qué punto. El ejecutar un benchmark para llevar al límite las capacidades de un sistema puede servir también en este sentido.

Comparación de diferentes programas que realizan una tarea determinada. Por ejemplo, evaluar cómo diferentes compiladores generan código, cómo se comportan dos sistemas de procesamiento de transacciones, o cómo aprovechan los recursos del sistema dos diferentes sistemas operativos.

Diseño de sistemas informáticos. Aunque, evidentemente, no se puede ejecutar un benchmark sobre un sistema que aún no se ha construido, la ejecución de un benchmark puede permitir extraer conclusiones sobre el comportamiento de un sistema para diseños futuros. Por ejemplo, el número de instrucciones por ciclo evaluados con diferentes benchmarks puede permitir extraer conclusiones para el diseño de un pipeline; o a partir de la tasa de aciertos de una caché se puede calcular cuál va a ser su tamaño en versiones sucesivas **(11)**, o el número de vías, o si se va a separar la caché de datos de la de instrucciones. De las misma forma, los benchmarks se pueden ejecutar a posteriori sobre un sistema informático para evaluar la efectividad de alguna de sus características en un entorno determinado, como hacen Bekerman y otros con el procesador Pentium, descubriendo que el pipeline V de enteros sólo se usa, como máximo, en el 30% de los casos, y mucho menos si se trata de programas comerciales en Windows. Incluso algunos autores, como Hennesy y Patterson han usado el comportamiento a bajo diferentes programas benchmark procedentes de SPEC92 como criterio para el diseño de una arquitectura, la DLX.

Como se ve, es necesario ejecutar benchmarks en todos los niveles funcionales de la arquitectura de un sistema informático **(12)**: microoperaciones, máquina convencional, sistema operativo, lenguajes orientados a problemas, y programas de aplicación; y durante diferentes etapas de la vida útil del mismo: diseño, prueba, y comercialización. Por supuesto, el tipo de benchmark elegidos, las métricas usadas sobre el mismo, y el análisis de los resultados será diferente en cada caso.

5.3 Utilización de un benchmark

Los benchmarks son usados por una gran cantidad de personas, sus resultados son también aprovechados por una amplia gama de profesionales, y, por tanto, la mayoría tienen cierto grado de **oficialidad o estandarización**. Las especificaciones para los mismos, las reglas para su ejecución y la ejecución física de los mismos se llevan a cabo de forma diferente.

5.3.1 Compañías que proponen un benchmark

Dado que los benchmarks sirven para comparar sistemas informáticos, deben de tener amplia difusión, y, de hecho, *todos* los sistemas informáticos deben ser evaluados por el mismo benchmark, para que efectivamente puedan ser comparados. Para que este objetivo se lleve a cabo, los benchmarks suelen ser ejecutados por una empresa o institución independiente, que habitualmente es pagada por sus servicios. La empresa recibe una copia del sistema a evaluar, realiza el trabajo necesario para que se ejecuten los programas, y entrega los resultados al fabricante. Esto, más o menos, garantiza la independencia de los resultados. En muchos casos, las empresas son revistas o están contratadas por revistas, que presentan los resultados obtenidos a los lectores. Este el caso de NSTL, National Standard Technology Labs, contratados por la revista Byte, o de las revistas del grupo Ziff-Davis, que recurren a la Ziff-Davis Benchmark Operation. En otros casos, son instituciones universitarias, como sucede en el caso del grupo de Investigación de Jack Dongarra, proponente de los benchmarks denominados LINPACK.

En este caso, la propia revista o institución suele llevar a cabo los benchmark, aunque también suele dejar en el dominio público los programas para que cualquiera pueda reproducir los resultados.

Ser propuestos por un consorcio de empresas, es decir, son una serie de programas, que todos o la mayoría de los fabricantes conoce y está de acuerdo en respetar los resultados; cada fabricante es responsable de llevar a cabo las pruebas y de publicar los resultados.

Esto sucede con SPEC, System Performance Evaluation Council, que propone benchmarks para medir diferentes aspectos de un sistema, o GPC, Graphics Performance Council, que hace lo propio para subsistemas gráficos, o TPC, Transaction Processing Council, que propone benchmarks para la medición de sistemas de proceso transaccional.

El propio interesado puede llevar a cabo los benchmarks, siempre que los diferentes sistemas informáticos estén disponibles y cuente con los recursos suficientes para hacerlo. De esta forma, se pueden enfocar claramente los objetivos del estudio, y usar las métricas y resultados que más interesen; además, tiene sentido que los usuarios conozcan como aplicar y qué suelen medir los benchmarks más importantes. Estas métricas se explican en el apartado siguiente.

5.3.2 Unidades de medida

Las unidades dependerán totalmente del objetivo del estudio y de qué nivel del sistema informático interese medir. Evidentemente, todas estas medidas reflejan las prestaciones de un sistema completo, puesto que es imposible separar los componentes para medirlos independientemente, aunque tratan de reflejar la eficiencia del sistema a este nivel. A continuación se enumeran las métricas usadas en diferentes niveles, aunque no todas se usan actualmente:

- A nivel de máquina convencional se pueden usar los MIPS (millones de instrucciones por segundo) o MFLOPS (millones de instrucciones en coma flotante por segundo). Estas medidas han caído en desuso, por su poca representatividad y porque es difícil que se reproduzcan resultados incluso para un mismo sistema, hasta el punto que se les conoce despectivamente como Meaningless o Marketing index of performance. Modernamente se usa también los CPI, o ciclos por instrucción, que mide el número medio de ciclos que tarda una instrucción en ejecutarse.
- En el nivel de sistema operativo se usan medidas tales como velocidad en la ejecución de un determinado programa y número de programas o usuarios concurrentes que tal sistema operativo es capaz de soportar.

- En el nivel de lenguajes orientados a programas, se mide el tamaño de un programa generado por un compilador, la eficiencia o velocidad de los programas generados por los mismos,
- A nivel de programas de aplicación, se mide como diferentes programas del mismo segmento son capaces de ejecutar la mismo tipo de carga de trabajo. Generalmente se miden velocidades.

Es habitual, en benchmarks, establecer un sistema informático como base de comparación, de forma que, en vez de dar las cantidades absolutas de las medidas, se den tales cantidades referidas al sistema base.

La mayoría de los índices presentados en las revistas de Informática y los consorcios de evaluación de prestaciones se hacen de esta forma.

En otros casos, el índice de prestaciones incluye el precio del sistema informático, de forma que se puedan comparar no sólo las prestaciones, sino también la relación precio/prestaciones en el caso de que el precio del equipo sea un factor a tener en cuenta.

Independientemente de las métricas que se extraigan del experimento, un benchmark mide prestaciones a todos los niveles funcionales del sistema informático, y por lo tanto todos influyen en los resultados. Desde el sistema operativo, hasta el compilador, pasando por las librerías ya compiladas que se usen para la ejecución de los programas.

Por ello hay que tener en cuenta y especificar cada una de ella junto con los resultados de la medición.

5.3.3 Modelos de benchmarking

Se puede afirmar de antemano que cuando una compañía dé un índice de prestaciones desnudo, sin indicar las condiciones en las que se ha llevado a cabo, está mintiendo.

El decir, por ejemplo, que un microprocesador puede ejecutar 22 MIPS, necesita ser cualificado por el programa o conjunto de programas con los que se alcanza tal medida. En cualquier caso, hay otras muchas formas de engañar al público con los resultados de un benchmark, lo cual se ha venido a denominar benchmarketing **(13)**:

- **Usar configuraciones diferentes para ejecutar la misma carga de trabajo**, por ejemplo distinta cantidad de memoria, o discos de diferente calidad o tamaño.
- **Elegir las especificaciones de forma que favorezcan a una máquina determinada,** generalmente, los sistemas están optimizados para una aplicación determinada; el probarlos con otra aplicación dará resultados incorrectos; y compararlos con respecto a esos resultados, también.
- **Usar una secuencia de trabajos sincronizada**, de forma que el solapamiento entre el trabajo de la CPU y del subsistema de E/S produzcan mejores prestaciones.
- **Elegir una carga de trabajo arbitraria,** que puede dar buenas prestaciones para una máquina determinada, pero no tener nada que ver con las prestaciones reales de la misma. Por ejemplo, escoger la "criba de Eratóstenes", o Whetstone, sin justificarlo en el tipo de trabajo para el cual se va a usar la máquina.
- **Usar benchmarks demasiado pequeños,** que pueden hacer que los fallos de página y de cache sean mínimos, de forma que se ignore la ineficiencia de la organización de las mismas. También pueden evitar los efectos de cambio de contexto y el overhead de la entrada/salida. La mayoría de los sistemas usan una amplia variedad de cargas de trabajo; para comparar dos sistemas, se deberían usar tantas cargas de trabajo como fuera posible.
- **Proyectar o interpolar** resultados de un benchmark **(14)**, es decir, medir las prestaciones de un ordenador con un procesador determinado, y proyectar los resultados a otro ordenador con un número de procesadores diferente, o un procesador de la misma familia con más potencia.

  Aunque se puede predecir el rendimiento ideal teniendo en cuenta la **ley de Amdahl** y la participación del nuevo procesador en las prestaciones del sistema, habitualmente los componentes de un sistema interaccionan de una forma tan compleja que es imposible usar esas proyecciones. Las prestaciones deben de medirse sobre una máquina real.

Tabla 5.1 Cuadro de comparación de resultados en 2 maquinas diferentes

	T(A)	T(A)/T(B)	T(B)	T(B)/T(A)
Benchmark 1	5	5	1	0.2
Benchmark 2	5	0.5	10	2
Media Aritmética	**5**	**2.25**	**5.5**	**1.1**
Media Geométrica	5	1.58	3.16	0.6

- **Elegir el sistema base de normalización de forma arbitraria**

Supóngase, por ejemplo, que al medir dos sistemas usando dos componentes de un benchmark se han obtenido los resultados de la tabla. En esta tabla 5.1, se presenta una situación bastante habitual, el que los resultados obtenidos en dos mediciones sean bastante dispares para una y otra máquina. Sin embargo, el usar una u otra máquina como índice puede resultar en que la máquina **B** es un 125% más rápida que la **A**, si se usa **B** como base, o que la **B** sea un 10% más rápida, si se usa la **A**. La media geométrica, sin embargo, mantiene la relación entre los tiempos de ejecución, sea cual sea la máquina que se toma de base, en este caso, **tiempo(A)/tiempo(B)==2/3**. Por esto, la media geométrica se suele usar en la mayoría de los benchmark estándar, como los benchmarks SPEC.

Sin embargo, en la mayoría de los casos, es difícil engañar con los resultados de los benchmarks, ya que ni el código de los mismos ni la presentación de sus resultados están bajo control de los fabricantes. Por eso lo más práctico es engañar directamente al benchmark, como se verá en la siguiente sección.

5.4 Algunos ejemplos de benchmarks

Hoy en día hay una serie de benchmarks, que están aceptados e incluso propuestos por la comunidad de fabricantes de ordenadores y estaciones de trabajo.

La mayoría están propuestos por consorcios, como los SPEC, TPC y GPC, que se verán en las siguientes secciones, aunque hay otros propuestos por editoriales de revistas del ramo, como el BYTEmark y el WinBench. En cada caso, se examinarán qué tipo de razonamiento ha conducido a elegir la carga de trabajo, las métricas usadas, y las críticas que se han vertido sobre ellos, algunas veces por los mismos fabricantes que los han propuesto.

5.4.1 Benchmarks propuestos por SPEC

SPEC, que son las iniciales de System Performance Evaluation Cooperative, es un consorcio de fabricantes de microprocesadores, ordenadores y estaciones de trabajo. La misión de SPEC es principalmente desarrollar una serie de programas que se van a utilizar para medir diversos aspectos de las prestaciones de un ordenador, y publicar los resultados de esos tests según han sido proporcionados por los fabricantes.

SPEC consiste en realidad en tres grupos diferentes:

- **OSG**, o Open Systems Group, que crea benchmarks para procesadores y sistemas que ejecutan UNIX, Windows NT y VMS.
- **HPC**, o High Performance Group, que mide prestaciones de ordenadores dedicados a cálculo intensivo, y
- **GPC**, o Graphics Performance Characterization Group, que mide prestaciones de subsistemas gráficos, OpenGL y XWindows.

SPEC solo especifica los programas que se van a utilizar para medir prestaciones y la forma de ejecutarlos, pero no los ejecuta, sino que confía en los fabricantes para ejecutarlos, con la condición de que los resultados serán supervisados por SPEC, y se presentarán en su página Web http://www.specbench.org. En realidad, dado el carácter público de los benchmark, cualquiera puede ejecutarlos (siempre que el ordenador cumpla las características que se especifican en la página Web, como 256 MBs de memoria y 1 GBs de disco).

Los benchmark de SPEC han tenido diversas versiones, la última de las cuales es la SPEC2000; las versiones anteriores son la 89, 92 y 95; siempre hay que tener en cuenta la versión a la hora de comparar prestaciones.

La mayoría de los fabricantes de ordenadores incluyen en sus páginas Web las medidas SPEC de sus equipos, eso sí, con las críticas correspondientes y poniendo claramente de relieve donde sobresalen. Incluso se basan en los resultados dados en estos benchmarks para tomar decisiones de diseño con respecto a sus máquinas.

a) **SPEC95 (Cint/Cfp)**

En concreto, SPEC95 se compone de dos grupos de programas: CINT95, 8 programas de cálculo de enteros intensivo, y CFP95, 10 programas de cálculo intensivo de coma flotante.

Tras medir lo que tardan en ejecutarse estos programas, se dan una serie de cantidades, algunas relacionadas con la velocidad, y otras relacionadas con el throughput, es decir, el número de programas del mismo tipo que un ordenador es capaz de manejar.

Todas las tasas son medias geométricas de los resultados obtenidos por cada uno de los diferentes métodos, no aritméticas. Estas medias geométricas se suelen utilizar cuando las prestaciones de los diferentes componentes no se suman, sino que se acumulan, como se verá más adelante. Y además, mantienen la relación entre los tiempos de ejecución de los benchmarks independiente de la máquina que se haya escogido para normalizar.

Por ejemplo, para SPECint, se dará lo indicado en la Tabla 5.2 siguiente

Tabla 5.2 Como seleccionar un tipo de SPEC

		Compilación	
		Conservador	Agresivo
Tasa	Velocidad	SPECint base95 SPECfp base95	SPECint95 SPECfp95
	Throughput	SPECint rate base95 SPECfp rate base95	SPECint rate95 SPECfp rate95

Se miden un total de 72 cantidades. Cada programa se compila de dos formas diferentes, una conservadora, con una serie de restricciones (por ejemplo, no más de 4 flags de ejecución), y además igual para todos los programas y máquinas, y otra agresiva, en la cual se puede usar la máxima optimización permitida por la máquina y el compilador; a la vez, para cada programa se mide el tiempo que tarda en ejecutarse, y el tiempo que tarda en ejecutarse cuando se ejecutan el máximo número de copias simultáneas del programa. Todas las velocidades se normalizan con respecto a un sistema, que depende de la versión del benchmark.

Tanto en el caso de Cint como de Cfp, los índices de "velocidad" suelen medir la eficiencia monoprocesador del sistema, aunque tenga diferentes procesadores; los índices de throughput miden la eficiencia del sistema operativo en cambios de contexto, pero también la eficiencia del mismo distribuyendo la carga entre diferentes procesadores, y la eficiencia del acceso a la memoria compartida de los mismos, en el caso de sistemas multiprocesador

Dado que los rangos afectan más al sistema operativo que a los resultados particulares de una compilación, y éste no cambia en los índices base y los agresivos, la inversión de prestaciones comentada anteriormente no se da en este caso.

Algunos fabricantes de ordenadores, a pesar de usar estas medidas, suelen criticar algunos de sus aspectos: la principal crítica es que la media oculta los resultados individuales de cada uno de los tests; uno de los tests puede dar mejor resultado para una máquina A que para otra B y sin embargo, la media sera mejor para B; por ello se presentan, en los informes de revelación total, los resultados individuales, para que cada persona pueda mirar los resultados que le interesan.

Estos tests miden también la combinación CPU/memoria/compilador, no el sistema completo, en el cual pueden influir mucho las prestaciones de entrada/salida o las de red; sin embargo, hay otros tests que pueden medir estos subsistemas; esto causa también que algunos fabricantes, como IBM, logren optimizar sus compiladores para que reconozcan en código SPEC y conseguir así mejores resultados.

Además, los "rate" miden las prestaciones de ordenadores ejecutando muchas copias del mismo programa, no diferentes programas, y por lo tanto son poco representativos de sistemas reales.

En general, hay que hacer un análisis pormenorizado de los resultados de cada uno de los benchmarks. En tal tipo de análisis, llevado a cabo por Sun, encontraron que el test más representativo es **gcc**, por tener código correspondiente a una aplicación real, un compilador usado en toda la gama de estaciones de trabajo, y ser además muy difícil de optimizar.

Por otro lado, **compress** es uno de los que más exige al sistema, haciendo énfasis en el ancho de banda entre CPU y memoria, que es esencial en todas las aplicaciones.

En estudios llevados a cabo por Intel, analizando cómo responden la jerarquía de memoria y el procesador (buffer TLB, caches, pipelines U y V) a estos benchmarks, halla también que **gcc** y **compress** tienen una baja localidad de las referencias, haciendo que el número de veces que se tiene que acceder a memoria principal es bastante alto.

Estudios similares se hacen en cada generación de las diferentes familias de microprocesadores, como por ejemplo en la tercera generación del PowerPC, denominada G3. En este modelo se incluyó un segundo pipeline de enteros tras examinar trazas de ejecución de diversos modelos anteriores.

b) SpecWEB99 y otros benchmarks de servidores web

SpecWEB99 (http://www.spec.org/osg/web99/) es la última versión de un benchmark que se concibió originalmente en el año 95; es un benchmark sintético para medir las prestaciones de sistemas como servidores web. Inicialmente incluía sólo páginas estáticas, pero hoy en día ha cambiado mucho. Tiene las siguientes características:

- Combina contenido estático y dinámico, pero se basa más en el estático que en el dinámico

- El contenido estático contiene ficheros de diferente tamaño y frecuencia decreciente; y dentro de cada "nivel", los ficheros tienen tamaños que se incrementan en un número fijo de Ks.

Por ejemplo, los del nivel de 100Ks se incrementan de 10 en 10 Ks. Se sigue una ley de Zipf (número de peticiones exponencialmente decreciente con el tamaño) para decidir qué fichero se pide el resultado principal de este benchmark es lo que se denomina "conexiones simultáneas conformantes", es decir, peticiones simultáneas que es capaz de manejar; se suele repetir tres veces y se da la mediana de los tres resultados como resultado válido.

Al principio, este benchmark recibió bastante atención por parte de los medios; por ejemplo, este artículo (http://linuxtoday.com/news_story.php3?ltsn=2000-07-05-001-04-OP) cuenta como se triplica prácticamente, sobre la misma máquina, las prestaciones de un servidor RH Linux sobre un servidor IIS. Sin embargo, mirando un poco de más cerca, vemos a qué se puede deber esta ventaja. El servidor que usa el sistema Red Hat es Tux, denominado actualmente Red Hat Content Accelerator, un servidor que se ejecuta como tarea en el kernel, en un sistema exclusivamente dedicado a eso (prácticamente). De hecho, casi todos los benchmarks enviados en el cuatrimestre actual (http://www.spec.org/web99/results/res2003q2/) incluyen este servidor, y, de hecho, es superior al competidor siguiente, el servidor web Zeus sobre un sistema IBM.

De hecho, el uso de este servidor web es relativamente irrelevante, ni siquiera aparece en el informe de servidores web. Por lo tanto, el hecho de que se use o no este servidor puede ser totalmente irrelevante para un caso real, donde los servidores van a ser probablemente el Apache o el IIS. Al parecer, la imposibilidad de superar a este servidor en los benchmarks ha hecho que la gente pase de ejecutarlo sobre sus máquinas, y probablemente conducirá a la revisión del mismo (igual que sucedió con el SpecWEB96, que tuvo que ser revisado por la introducción por parte de Sun del Network Caché Accelerator, un módulo del kernel que metía las páginas estáticas en un buffer de la memoria del kernel).

August 2003

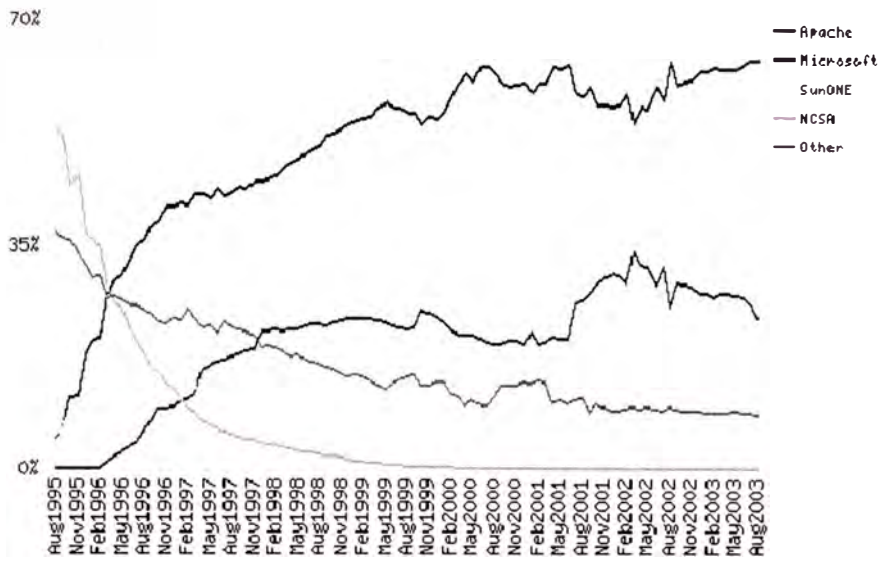


Figura 5.1 Participación de Mercado de algunos servidores Web August 1995 –

Tabla 5.3 Cuadro comparativo de servidores Web

Developer	July 2003	Percent	August 2003	Percent	Change
Apache	26951879	63.72	27388860	63.98	0.26
Microsoft	10976342	25.95	10165745	23.75	-2.20
SunONE	674571	1.59	1534586	3.58	1.99
Zeus	766943	1.81	746240	1.74	-0.07

5.4.2 TPC

TPC (Transaction Processing Performance Council) es un consorcio que propone benchmarks en el área de procesamiento de transacciones. Hay varios benchmarks propuestos; hoy en día los más usados son TPC-C y TPC-D. Ambos tienen como entorno de referencia el siguiente: un sistema cliente, que genera peticiones de bases de datos usando un front-end, y un sistema servidor, que ejecuta el intérprete del lenguaje de bases de datos, generalmente SQL, y desde el mismo accede a un sistema de gestión de bases de datos. Cada transacción debe de ser culminada, es decir, se lleva a cabo o no completa, y además los sistemas deben de estar disponibles 24x7, 24 horas al día 7 días a la semana.

Estos benchmarks son de muy alto nivel, o más bien, ejercitan todos los niveles del sistema, desde la jerarquía de memoria, incluyendo memoria secundaria, pasando por el sistema operativo, hasta los programas que resuelven un problema concreto.

Las métricas que se proporcionan con estos benchmarks son de dos tipos: peticiones/tiempo, número de usuarios, y precio/prestaciones; hay unas reglas estrictas para evaluar éste último. Se verá a continuación cada uno de los benchmarks por separado.

a) TPC-C

Heredero de TPC-A y B, en este benchmark se mide el número de transacciones New Order que se pueden generar por minuto mientras que un sistema está ejecutando peticiones de otros cuatro tipos (Pago, Estado de la Petición, Reparto, Nivel de Stock). Las cinco peticiones deben de llevarse a cabo en un tiempo máximo determinado; New Order debe de durar 5 segundos como máximo.

Las métricas que se usan son tpmC (transacciones por minuto de tipo C) y $/tmpC, dividiendo las mismas por el precio total del sistema.

b) TPC-D

Similar al anterior, salvo que modela un sistema de apoyo a la decisión, lo que se ha dado en llamar data warehousing o data mining, es decir, peticiones complejas a bases de datos de gran tamaño. En este tipo de entornos suele haber más peticiones de lectura que de escritura, y además se suele acceder a una gran porción de la base de datos en una sola petición.

El entorno de experimentación consiste en responder a 17 peticiones, en diferente orden, mientras que en segundo plano se efectúan operaciones de inserción y borrado. Las órdenes están codificadas en SQL. Las bases de datos a las que se accede tienen diferentes factores de escala, indicados con una @; al proporcionar las prestaciones hay que indicar el factor de escala, y no se pueden extrapolar los resultados de un factor de escala a otro superior o inferior.

Los resultados obtenidos por TPC tienen valor de certificación; generalmente suelen ser bastante onerosos para una empresa, y a veces incluyen una auditoria por parte de una empresa independiente.

TPC-D tiene tres métricas, dos de prestaciones y una de precio/prestaciones:

- QppD@tamaño, o métrica de potencia, media geométrica del tiempo empleado por las 17 peticiones, la inserción y el borrado.
- QthD@tamaño, una medida de tipo throughput, que caracteriza la habilidad del sistema para soportar una carga de trabajo multiusuario de tipo equilibrado. Se elige un número de usuarios S, cada uno de los cuales ejecuta las 17 peticiones en orden diferente; la inserción y el borrado también se llevan a cabo en segundo plano. El número de usuarios lo elige el que paga la realización del test (sponsor) S puede ser 1.
- QphD@tamaño, media geométrica de las dos métricas. Su principal propósito es dar un solo índice con el cual se pueda calcular la relación precio/prestaciones.

De esta última métrica, se obtiene una métrica precio prestaciones. Por ejemplo, si el coste de un sistema es 5 millones de dólares, se ha tomado las mediciones para una base de datos de 100 GB, y se ha obtenido un índice QphD@100GB de 141.42, la métrica precio-prestaciones será 35.335'34$/QphD@100 GB.

Dada la importancia comercial que tienen estos benchmarks, hay algunas empresas que han vertido críticas sobre ellas, atendiendo sobre todo a que no representan realmente la carga de trabajo generada por un sistema de toma de decisiones. Otra crítica es que son demasiado caros, costándole a una empresa el orden de un millón de dólares. Por tanto, algunas empresas proponen medir también el proceso de carga, la realización de consultas y los procesos de análisis.

5.4.3 Resumen de benchmark

En el cuadro siguiente se resumen los benchmarks comentados, de qué tipo son, quién los ha propuesto, quién los ejecuta y el sistema o subsistema que ejercitan.

Tabla 5.4 Comparación de uso entre SPEC y TPC

Benchmark	Propuesto por	Tipo	Subsistema/función	Ejecutado por
SPEC CPU	SPEC	Artificial Sintético	CPU/Memoria/compilador	Fabricante
SPEC SFS		Artificial Sintético	Sistema de ficheros UNIX	
SPECWeb		Artificial Sintético	Sistema como servidor Web	
SPEChpc		Smtético Natural	Sistema completo en cálculo de altas prestaciones	
TPC-C	TPC	Artificial Sintetico	Sistema completo para proceso de transacciones	Fabricante supervisado por un auditor
TPC-D		Artificial Sintético	Sistema completo para toma de decisiones	

CONCLUSIONES

1. En base al estudio realizado con respecto al diseño de cada microprocesador de 64 bits podemos establecer algunas diferencias en cuanto a su eficacia y desempeño.

2. En el caso de los Microprocesadores Sun Microsystems encontramos una tecnología basada principalmente en un sistema de arreglo de buses de comunicación entre los CPUs y los dispositivos externos, lo cual mejora y facilita un rendimiento optimo cuando se trata con información que puede ir cambiando en el tiempo, para el caso de los Microprocesadores IBM encontramos en cambio una tecnología basada en un mejor desempeño a nivel de procesamiento en el interior del CPU, el cual cuenta con un mayor ancho de bus de información y permite ejecutar mayor cantidad de instrucciones en cada ciclo de reloj. Cabe notar que ambos aprovechan dentro de sus CPUs el direccionamiento a una gran memoria que les permite incrementar su velocidad de procesamiento, los 64bits que utilizan ambos procesadores permite aprovechar el uso de aplicaciones desarrolladas bajo este tipo de plataforma, a diferencia de Sun el microprocesador de IBM permite correr aplicaciones nativas de 32 bits sin tener que modificar programas esto es una ventaja a la hora de evaluar costos para migrar sistemas, siendo un paso previo que permite adecuar las aplicaciones a la nueva plataforma.

3. Ambos microprocesadores están basados en tecnología CMOS, lo cual les permite consumir poca energía, y ser construidos cada vez más pequeños, en el caso de Sun Microsystems su tecnología le permite tener hasta 7 capas (6 en cobre y 1 en aluminio) de interconexión por CPU en el caso de los microprocesadores IBM hasta 8 capas (todas en cobre) de interconexión por CPU.

4. Uno de los puntos fuertes de la tecnología IBM es sin lugar a dudas la velocidad de su Bus de comunicaciones el cual opera a 900Mhz y con un acho de 64bits lo cual le permite una gran rapidez para la ejecución de diferentes procesos hacia los demás dispositivos , en el caso de Sun Microsystems este ha desarrollado un chip cache para el bus el cual le permite compartir la información con los demás microprocesadores dentro de un arreglo con 4 CPUs, su velocidad de operación es de 200Mhz pero con un ancho de 128bits.

5. A simple vista es difícil discernir cual microprocesador es mejor dado que hoy en día existen una serie de aplicaciones que pueden explotar en mejor medida determinadas características de un microprocesador, dicha incertidumbre nos obliga a usar métodos comerciales y algunos no convencionales para medir y ver que equipo nos podría resultar mas favorable en nuestras aplicaciones, de acuerdo a lo que desarrollamos en el capitulo III Herramientas de Monitoreo, se expuso de manera sencilla como utilizar métodos comerciales económicos para evaluar en forma gráfica y visual el comportamiento de diferentes partes de nuestro sistema con respecto a una aplicación, dentro de este método se pudo apreciar dos tipos, uno por hardware y otro por software.

6. Las herramientas de monitoreo de Hardware, no interfieren en el sistema ya que por lo general se consiguen los datos con el uso de equipos de medición, que hacen lecturas del comportamiento de algún subsistema en particular (Discos, canales de I/O, CPU, etc).

7. Por lo general, las herramientas de Software, afectan al sistema en observación ya que requieren ser instalados bajo el sistema operativo en observación, consumiendo de él recursos importantes, como ciclos de CPU, memoria y degradan la performance al exigir los buses de I/O para grabar los registros con la información capturada. Estos monitores son de dos tipos: "event-driven" o "samplers" (a la carrera o por muestras); los primeros suelen consumir mas recursos que los otros ya que requieren estar habilitados todo el tiempo, produciendo "overhead" al sistema observado; mientras que los muestreadotes se activan a intervalos pre-definidos por el usuario.

8. En ambos casos podemos obtener una medición cercana a la realidad y nos puede ayudar en un indicador aceptable para dimensionar nuestra infraestructura.

9. El otro método más comercial pero con cierta maniobra por parte de los fabricantes es el uso de las Herramientas de medición o Benchmark, los cuales están basados en organismos e instituciones internacionales que se dedican a medir el rendimiento y la performance de microprocesadores, estaciones de trabajo y servidores buscando su máxima productividad con pruebas de diversas aplicaciones.

10. Dicha información normalmente es de carácter público en la web porque ayuda a determinar el posicionamiento de una marca, el problema al igual que todo lo relacionado con tecnología es que cada cierto tiempo dicha información tiene que ser cambiada porque siguen saliendo nuevos microprocesadores con nuevas características y que traen consigo mejoras en rendimiento. Las marcas de mayor prestigio utilizan esta información para posicionar en el mercado sus productos, así podemos encontrar en ella a Intel, IBM, HP, Sun, AMD, etc.

11. En la mayoría de los casos la supremacía esta dada por las capacidades de procesamiento en minuto, donde encontramos muchas veces equipos sobredimensionados que escapan a la realidad de las empresas o compañías. Solo nos sirven como dato relevante de cuanto puede crecer un equipo con menores características y de cuanto podría soportar de requerir una demanda similar. Pero como sabemos la tecnología cambia y cuando pensamos que nuestro equipo nos puede ayudar a crecer a futuro nos encontramos de que dicho equipo en unos cuantos años ya es obsoleto y nuevamente regresamos a buscar algo que se adecue a las nuevas realidades.

ANEXO A

Información sobre los bancos de pruebas internacionales para el cualquier procesador

TPC-C: El banco de pruebas TPC-C ha sido diseñado con el propósito de medir la capacidad de un servidor para funcionar como servidor de base de datos para el procesamiento de transacciones on-line (OLPT). Este banco de pruebas simula un entorno completamente informático donde los usuarios contrastan las transacciones de los registros con la base de datos. Entre estas transacciones se incluyen órdenes de compra, realización o recepción de pagos, seguimiento de las órdenes de compra, monitorización de la entrega y control de los niveles de stock. El resultado que ofrece TPC-C es una medición del número de nuevas órdenes de compra generadas por minuto, mientras que el sistema ejecuta de forma simultánea los otros cuatro tipos de transacciones.

MMB2: El banco de pruebas MAPI Messaging Benchmark (MMB2) fue desarrollado por Microsoft con el propósito de medir la capacidad de un servidor para funcionar como plataforma de servidor Microsoft Exchange. MMB2 mide el rendimiento del "Usuario Medio" que ejecuta tareas comunes como navegar por la red, enviar y recibir mensajes de correo electrónico, así como programar tareas y utilizar las listas de distribución, en una jornada laboral de ocho horas.

SPECweb®99: SPECweb99 ha sido diseñado para medir la capacidad de un sistema para funcionar como servidor Web para páginas estáticas y dinámicas. En la configuración del banco de pruebas, cierto número de sistemas cliente generan las peticiones de páginas estáticas y dinámicas en lugar de un servidor, simulando la carga de trabajo de un servidor Web real. Los resultados obtenidos con el banco de pruebas SPECweb99 representan el número máximo de conexiones conformadas simultáneas que un servidor Web puede soportar sin disminuir los requisitos específicos de rendimiento e índice de error. Las conexiones conformadas deben estar establecidas con un índice máximo de bits y con un tamaño máximo de segmento. El propósito de esto es recrear las condiciones que se producirán en Internet durante la vida útil de este banco de pruebas.

SPECweb99_SSL: Al igual que SPECweb99, SPECweb99_SSL ha sido diseñado con el propósito de medir la capacidad de un sistema para funcionar como servidor Web para las páginas estáticas y dinámicas. No obstante, la diferencia de SPECweb99_SSL se encuentra en que las páginas han sido solicitadas utilizando HTTP a través de Secure Sockets Layer Protocol (HTTPS), por lo tanto, los resultados obtenidos con el banco de pruebas se emplean con el objetivo de determinar la capacidad de un sistema para funcionar como servidor Web seguro.

SPECjbb™2000: SPECjbb2000 ha sido diseñado con el objetivo de medir la capacidad de un sistema para funcionar como servidor de aplicaciones java de nivel medio en un sistema de 3 niveles, que se compone de un cliente (nivel 1), un servidor de aplicaciones Java (nivel 2) y una base de datos (nivel 3).

El servidor de aplicaciones Java contiene las caches lógica y de objetos de la empresa, que gestionan las solicitudes entrantes del cliente, recuperan la información apropiada de la base de datos y devuelven esa información al cliente. Estos servidores se suelen utilizar en ERP, CRM, e-Business y demás aplicaciones de nivel que emplean un navegador Web para acceder a la información almacenada en la base de datos. El banco de pruebas SPECjbb2000 se ha desarrollado a partir de una empresa mayorista con almacenes que distribuyen en varios distritos.

El resultado de SPECjbb2000 mide el rendimiento de la plataforma Java subyacente, que es el índice de ejecución por segundo, de ciertas actividades empresariales.

SPECint®2000: SPECint2000 ha sido diseñado para medir y comparar el rendimiento completo entre sistemas que requieren una gran cantidad de recursos. Este banco de pruebas se centra en el rendimiento de:

- El procesador (CPU)
- La arquitectura de memoria
- Los compiladores

SPECint2000: se compone de doce bancos de pruebas enteros que han sido desarrollados a partir de las actuales aplicaciones de los usuarios finales. Las aplicaciones pesadas suelen ser las más habituales en los departamentos de TI y en los servidores empresariales.

Normalmente, aplicaciones como las bases de datos, servidores de correo electrónico, servidores de aplicaciones Java y servidores Web, suelen ejecutarse mejor en un procesador con un excelente rendimiento completo.

SPECint_rate™2000: Este banco de pruebas funciona con los mismos algoritmos utilizados en SPECint2000, pero ejecuta múltiples ejemplos del banco de pruebas al mismo tiempo (normalmente se ejecuta un ejemplo por procesador del sistema). SPECint_rate2000 determina la capacidad de un sistema para realizar al mismo tiempo, múltiples operaciones enteras que requieren una gran cantidad de recursos. Este banco de pruebas ha sido diseñado con el propósito de medir la capacidad de un sistema con varios procesadores para escalar adecuadamente, mientras se ejecutan aplicaciones como los servidores de base de datos, servidores de correo electrónico o servidores Web.

SPECfp®2000: SPECfp2000 ha sido diseñado para medir y comparar el rendimiento intensivo de coma flotante entre sistemas. Este banco de pruebas se centra en el rendimiento de:

- El procesador (CPU)
- La arquitectura de memoria
- Los compiladores

SPECfp2000: se compone de catorce bancos de pruebas de coma flotante que han sido desarrollados a partir de las aplicaciones de los usuarios finales. Las aplicaciones intensivas de coma flotante son las más habituales en los entornos de ingeniería e investigación. Las aplicaciones informáticas para la dinámica de fluidos, CAD/CAM, creación de contenidos digitales (DCC) representación, así como las herramientas financieras, normalmente funcionan mejor cuando se ejecutan en un procesador con un excelente rendimiento de coma flotante.

SPECfp_rate2000: Este banco de pruebas funciona con los mismos algoritmos utilizados en SPECfp2000, pero ejecuta múltiples ejemplos del banco de pruebas al mismo tiempo (normalmente se ejecuta un ejemplo por procesador del sistema). SPECfp_rate2000 determina la capacidad de un sistema para realizar al mismo tiempo, múltiples operaciones de coma flotante que requieren una gran cantidad de recursos. Este banco de pruebas ha sido diseñado con el propósito de medir la capacidad de un sistema con varios procesadores para escalar adecuadamente, mientras se ejecutan aplicaciones basadas en coma flotante como CAD/CAM, DCC y otras aplicaciones científicas.

BIBLIOGRAFÍA

1. D. Patterson and J. L. Hennessy, Computer Organization and Design: The Hardware/ Software Interface, 2nd Edition, Morgan Kaumann Publisher, Inc., San Franciso, California, 1997.

2. K. Normoyle, "A Dash of Jalapeno: Introducing the UltraSPARC IIIi Microprocessor", Microprocessor Forum, 2001.

3. R. N. Raman, "JBus: A Bus Architecture for Embedded On-chip Multiprocessing", Embedded Microprocessor Forum, 2001.

4. Sun's Throughput Computing Web page (www.sun.com/ultrasparc/throughput)

5. Sun Microsystems. (2001). mediaLib User's Guide.
Bhaskaran, V. and Konstantinides, K. (1997). Image and Video Compression Standards. Kluwer Academic Publishers.

6. PowerPC Microprocessor Family: AltiVec Technology Programming Environments Manual

7. D. E. Lenoski andW.D.Weber, Scalable Shared-Memory Multiprocessing, Morgan Kaumann Publisher, Inc., San Franciso, California, 1995.

8. http://www-306.ibm.com/chips/techlib/techlib.nsf/techdocs/

9. IBM eServer BladeCenter Configuration Tips, TIPS-0454

10. B. P. Sinha and P. K. Srimani, Fast Parallel Algorithms for Binary Multiplication and Their Implementation on Systolic Architectures, IEEE Transactions on Computers, vol. 38, No. 3, 424-431, March 1989

11 [Franklin91] J. Franklin, Fuzzy Representations in Neural Nets, 1991, Fuzzy Logic and Fuzzy Control, D. Driankov and P. W. Eklund and A. L. Ralescu, Springer,

12. [Tanenbaum87] Tanenbaum A. Operating Systems: Design and lmplementation. Prentice-Hall, 1987

13. [Jain91] Raj Jain, The Art of Computer Systems Performance Analysis: Techniques for experimental Design, Measurement, Simulation, and Modeling, John Wiley & Sons, (1991)

14. [Bailey91], David H. Bailey Twelve Ways to Fool the Masses When Giving Performance Results on Parallel Computers, Supercomputing Review 1991