

Organización de las computadoras
Resumen para rendir final – plan 2003

Contenido

1. Computadoras digitales	4
1.1 Conceptos introductorios	4
1.2 Funcionamiento básico	4
1.3 Estructura	5
1.4 Evolución histórica de las computadoras y la tecnología empleada en su fabricación	6
1.5 Diseño para conseguir mejores prestaciones	10
1.6 Evolución del Pentium (CISC) y del Power PC (RISC).	12
1.7 Definiciones varias	13
1.7 Niveles de abstracción	15
2. Aritmética de las computadoras	16
2.1 Definición de bit, nibble, byte, palabra, relación con lenguajes de alto nivel	16
2.2 Teorema Fundamental de la Numeración:.....	17
2. 3 Sistema Octal	18
2.4 Representaciones numéricas: números enteros con y sin signo	18
2.5 Aritmética con enteros.....	20
2.6 Fundamentos de la representación en punto flotante, normalización, error de la representación.....	21
2.7 Representación estándar del IEEE.....	23
2.8 Operaciones aritméticas en punto flotante.....	24
2.9 Representaciones alfanuméricas, ASCII, EBCDIC.....	27
3. Lógica Digital	28
3.1 Compuertas lógicas	28
3.2 Álgebra de Boole	29
3.3 Implementación de funciones booleanas	30
3.4 Lógica combinatoria, codificadores, decodificadores, multiplexores	30
3.5 Lógica secuencial, registros, contadores	35
3.6 Aplicaciones de lógica digital en la Unidad Aritmético - Lógica (ALU) y en la Unidad de Control.	42
4. Unidad Central de Procesamiento (CPU)	43
4.1 Organización de la CPU	43
4.2 Características de las instrucciones máquina.....	44
4.3 Flujo de datos	45
4.4 Tipo de instrucciones.....	46
4.5 Lenguaje de máquina y ensamblador	48
4.6 Organización de los registros	51
4.7 Direccionamiento.....	52
4.8 Formato de instrucciones.....	56
4.9 El ciclo de instrucción	57
4.10 Interrupciones	59
4.11 Estructuras de interconexión	60
4.12 Descripción de microprocesadores actuales	63
5. Memoria	65
5.1 Conceptos básicos sobre sistemas de memoria de computadores	65
5.2 Jerarquía de memoria.....	66
5.3 Localidad de referencias	67
5.3 Memoria principal semiconductora	69
5.4 Memoria caché	72

Organización de las computadoras
Resumen para rendir final – plan 2003

5.5 Memoria externa.....	72
5.6 RAID	74
5.7 Memoria óptica.....	79
6. Periféricos.....	81
6.1 Comunicación hombre-máquina.....	81
6.2 Comunicación máquina-mundo físico.....	85
6.3 Comunicación máquina-máquina	89

1. Computadoras digitales

Conceptos introductorios. Funcionamiento básico. Organización de un sistema de cómputo, modelo de von Neumann. Otros modelos de organización, clasificación de las computadoras de acuerdo al modelo. Ejemplos. Evolución histórica de las computadoras y la tecnología empleada en su fabricación. Costo y rendimiento. Análisis de la performance, métodos de medición, MIPS, MFLOPS, benchmarks. Concepto de niveles de abstracción.

1.1 Conceptos introductorios

La **arquitectura** de computadores se refiere a los atributos de un sistema que son visibles para un programador, o, para decirlo de otra manera, a aquellos atributos que tienen un impacto directo en la ejecución lógica de un programa. La **organización** de computadores se refiere a las unidades funcionales y sus interconexiones, que dan lugar a especificaciones arquitectónicas. Entre los atributos de organización se incluyen aquellos detalles de hardware transparentes al programador tales como señales de control, interfaces entre el computador y los periféricos y la tecnología de memoria usada.

Para dar un ejemplo, una cuestión de diseño arquitectónica es si la CPU tendrá la instrucción de multiplicar. Una cuestión de organización es si esta instrucción será realizada por una unidad especializada en multiplicar o si se hará por sumas sucesivas. Una arquitectura puede perdurar muchos años, pero la organización cambia con el avance de la tecnología.

1.2 Funcionamiento básico

Las funciones básicas que un computador puede llevar a cabo son, en términos generales, cuatro:

1. **Procesamiento de datos:** el procesador tiene que ser capaz de procesar datos. Los datos pueden adoptar una gran variedad de formas, y el rango de los requisitos de procesado es amplio. Sin embargo, hay sólo unos pocos métodos o tipos fundamentales de procesado de datos.
2. **Almacenamiento de datos:** también es esencial que un computador almacene datos. Incluso si el computador está procesando datos al vuelo (es decir, si los datos se introducen y se procesan y los resultados se obtienen inmediatamente), el computador tiene que guardar temporalmente, al menos aquellos datos con los que está trabajando en un momento dado. Con igual importancia, el computador lleva a cabo una función de almacenamiento de datos a largo plazo.
3. **Transferencias de datos:** el computador debe ser capaz de transferir datos entre él y el mundo exterior. Cuando se reciben o se llevan datos a un dispositivo que está directamente conectado con el computador, el proceso se conoce como E/S y el dispositivo recibe el nombre de periférico. El proceso de transferencia de datos a largas distancias, se lo conoce con el nombre de comunicación de datos.

4. Control: finalmente, debe haber un control de las últimas tres funciones. Este control es ejercido por los entes que proporcionan al computador instrucciones. Una unidad de control gestiona los recursos del computador y dirige las prestaciones de sus partes funcionales en respuesta a las instrucciones.

1.3 Estructura

Un computador es un sistema complejo: los computadores de hoy en día contienen millones de componentes electrónicos básicos. Por esta razón, los computadores se organizan como un sistema jerárquico.

Un sistema jerárquico es un conjunto de subsistemas interrelacionados, cada uno de los cuales, a su vez, se organiza en una estructura jerárquica, hasta que se alcanza el nivel más bajo de subsistemas esenciales. Esta organización es esencial, tanto para el diseño del computador, como para su descripción. El diseñador necesita tratar con un nivel a la vez. El comportamiento en cada nivel depende sólo de una caracterización abstracta y simplificada del sistema que hay en el siguiente nivel más bajo.

El computador es una entidad que interactúa de algún modo con el mundo exterior. En general, todas sus conexiones con el entorno externo pueden ser clasificadas como *periféricos* o *líneas de comunicación*. Hay cuatro componentes estructurales básicos, aunque puede que haya uno o más de cada uno de ellos:

- **Unidad central de procesamiento - CPU:** controla el funcionamiento del computador y lleva cabo sus funciones de procesamiento de datos.
- **Memoria principal:** almacena datos.
- **E/S:** transfiere datos entre el computador y el mundo exterior.
- **Sistemas de interconexión:** es un mecanismo que proporciona la comunicación entre la CPU, la memoria y la E/S.

A su vez, la CPU se puede dividir en cuatro elementos:

- **Unidad de Control:** controla el funcionamiento de la CPU y, por lo tanto, de todo el computador.
- **Unidad Aritmético-Lógica:** lleva a cabo las funciones de procesamiento de datos del computador.
- **Registros:** proporcionan almacenamiento interno a la CPU.
- **Interconexiones CPU:** son mecanismos que proporcionan comunicación entre la unidad de control, la ALU y los registros.

1.4 Evolución histórica de las computadoras y la tecnología empleada en su fabricación.

1946: La primera generación - los tubos de vacío

ENIAC (Electronic Numerical Integrator And Computer)

Fue el primer computador electrónico de propósito general del mundo, construido usando tubos de vacío. Pesaba 30 toneladas, ocupaba 15 mil pies cuadrados y contenía más de 18 mil tubos de vacío.

El ENIAC era una máquina decimal, no binaria. Su memoria consistía en 20 acumuladores, cada uno capaz de contener un número decimal de 10 dígitos. Cada dígito estaba representado por un anillo de 10 tubos de vacío. En un momento dado, sólo uno de los tubos de vacío estaba en estado ON, representando uno de los 10 dígitos. Uno de los mayores inconvenientes del ENIAC era que tenía que ser programado manualmente mediante conmutadores y conectando y desconectando cables.

La máquina de Von Neumann

En la máquina ENIAC la carga y modificación de programas era extremadamente tediosa. El proceso de programación podría ser más fácil si el programa se representara en una forma adecuada para ser guardado en la memoria, junto a los datos. Esta idea, conocida como concepto de *programa almacenado*, se atribuye a los diseñadores del ENIAC, sobre todo al matemático Neumann, que era el asesor del proyecto ENIAC. La primera publicación de la idea fue en la de von Neumann para un nuevo computador en 1945, el EDVAC.

En 1946, von Neumann y sus colegas empezaron el diseño del IAS. No completado hasta 1952, contenía el concepto de programa almacenado y es el prototipo de toda una serie de computadores de propósito general.

IAS constaba de una memoria principal, que almacena tantos datos como instrucciones; una ALU, capaz de hacer operaciones con datos binarios; una unidad de control, que interpreta las instrucciones en memoria y provoca su ejecución y un equipo de E/S, dirigido por la unidad de control.

Salvo raras excepciones, todos los computadores de hoy en día tienen la misma estructura general y funcionamiento que la indicada en las máquinas de von Neumann.

La memoria IAS consistía en 1000 posiciones de almacenamiento, llamadas palabras, de 40 dígitos binarios cada una. Tanto los datos como las instrucciones se almacenaban en ellas. Los números y las instrucciones se representaban en forma binaria. Cada número se representa con 1 bit de signo y 39 de valor. Una instrucción podía contener dos instrucciones de 20 bits, donde cada instrucción consistía en un código de operación de 8 bits (*codop*), que especificaba la operación que se iba a realizar, y una dirección de 12 bits, que indicaba una de las palabras de memoria.

La unidad de control dirigía el IAS captando instrucciones de la memoria y ejecutándolas una por una. Tanto la ALU como la unidad de control contenían posiciones de almacenamiento propias, llamadas registros, definidos de la siguiente manera:

- **Registro temporal de memoria (MBR):** contiene una palabra que debe ser almacenada en memoria, o usada para recibir una palabra procedente de la memoria.
- **Registro de dirección de memoria (MAR):** especifica la dirección en memoria de la palabra que va a ser escrita o leída en MBR.
- **Registro de instrucción (IR):** contiene los 8 bits del codop que se va a ejecutar.
- **Registro temporal de instrucción (IBR):** empleado para almacenar temporalmente las instrucciones contenidas en la parte derecha de la memoria.
- **Contador de Programa (PC):** contiene la dirección de la próxima instrucción que va a ser captada de la memoria.
- **Acumulador (AC) y multiplicador cociente (MQ):** se emplea para almacenar operandos y resultados de operaciones de la ALU temporalmente.

El IAS operaba ejecutando repetidamente un ciclo de instrucción. Cada ciclo consistía en varios subciclos.

Durante el ciclo de *captación*, el codop de la siguiente instrucción a ejecutar es cargado en el IR y la parte que contiene la dirección es almacenada en el MAR. Esta instrucción puede ser captada desde el IBR, o puede ser obtenida de la memoria cargando una palabra en el MBR, y luego en IBR, IR y MAR. Para simplificar la electrónica, se usa un solo registro para especificar la dirección en memoria para la lectura o escritura, y un solo registro para la fuente o el destino.

Una vez que el codop está en IR, se lleva a cabo el ciclo de *ejecución*. Los circuitos de control interpretan el codop y ejecutan la instrucción, enviando señales de control adecuadas para provocar que los datos se transfieran o que la ALU realice una operación.

El computador IAS tenía un total de veintiún instrucciones. Se podrían agrupar de la siguiente manera:

- **Transferencia de datos:** transferir datos entre la memoria y los registros de la ALU o entre dos registros de la ALU.
- **Salto incondicional:** la unidad de control ejecuta instrucciones secuencialmente en la memoria. Los saltos ocasionan el cambio de esta secuencialidad.
- **Salto condicional:** el salto depende de una condición, lo que permite puntos de decisión.
- **Aritmética:** operaciones realizadas por la ALU.
- **Modificación de direcciones:** permite que la ALU haga operaciones con las direcciones y las inserte en instrucciones almacenadas en memoria. Esto permite una considerable flexibilidad de direccionamiento en un programa.

1947: Computadores comerciales

UNIVAC 1 fue la primera computadora de uso comercial, aunque también tenía aplicaciones científicas. Le sucedió **UNIVAC 2**, que tenía una capacidad de memoria mayor y más aplicaciones.

Por otro lado, IBM sacó su primer computador con programas almacenados electrónicamente en 1953, el 701, diseñado para aplicaciones científicas.

1950: La segunda generación - los transistores

Los **transistores** sustituyeron a los tubos de vacío debido que son más pequeños, baratos, disipan menos el calor y pueden ser usados de la misma forma que un tubo de vacío en la construcción de computadores. Mientras que un tubo de vacío requiere cables, placas de metal, una cápsula de cristal y vacío, el transistor es un dispositivo de estado sólido, hecho con silicio.

En la segunda generación se introdujeron unidades lógicas y aritméticas y unidades de control más complejas, así como el uso de lenguajes de programación de alto nivel, y se proporcionó un software del sistema con el computador. La segunda generación también es destacable por el desarrollo del PDP-1, en 1957, de gran importancia en la tercera generación.

El computador más representativo de esta generación es el IBM 7094. Éste tiene varias diferencias con el computador IAS. La más importante es el uso de canales de datos. Un canal de datos es un módulo de E/S independiente, con su propio procesador y su propio conjunto de instrucciones. Esta disposición libera a la CPU de una carga de procesamiento considerable. Otra característica es el multiplexor, que es el punto de conexión central de los canales de datos, la CPU y la memoria. El multiplexor organiza los accesos a la memoria desde la CPU y los canales de datos, permitiendo a estos dispositivos actuar de forma independiente.

1958: La tercera generación - los circuitos integrados

Los circuitos integrados utilizaron el hecho de que los componentes usados en la segunda generación podían ser fabricados como un **circuito entero** a partir de un trozo de un semiconductor como el silicio y formar chips. Cada chip consiste de muchas puertas, más una serie de puntos para conexiones de E/S. El chip es encapsulado en una carcasa que lo protege y proporciona patas para conectar dispositivos fuera del chip. Varios de estos elementos pueden ser interconectados en una tarjeta de circuitos más complejos y mayores.

1964 - IBM 360

El **Sistema/360** fue la primera familia de computadores de la historia que se planeó. La familia abarcaba un amplio rango de prestaciones y precios. Los distintos modelos eran compatibles en el sentido de que, un programa escrito para un modelo,

tenía que ser capaz de ser ejecutado por otro modelo de la serie, con la única diferencia del tiempo de ejecución.

Características de una familia son:

- **Conjunto de instrucciones similar o idéntico:** así, un programa que se ejecuta en una máquina, se podrá ejecutar en cualquier otra. Los programas se pueden mover hacia arriba, pero no hacia abajo, debido a que el computador más bajo de la familia tienen un conjunto de instrucciones que es un subconjunto del computador más alto de la familia.
- **Sistemas operativos similares o idénticos.**
- **Velocidad creciente.**
- **Número creciente de puertos de E/S.**
- **Tamaño de memoria creciente.**
- **Costo creciente.**

1964 - DEC PDP-8

Es importante por dos características: su bajo costo (16 mil dólares) y su pequeño tamaño (entraba en una mesa de laboratorio) que le dio el nombre de “minicomputador”.

En contraste con la arquitectura del conmutador central usada por IBM en sus sistemas 700/7000 y 360, los últimos modelos del PDP-8 usaban una estructura que ahora es universal: el bus. El bus PDP-8, llamado Ómnibus, consiste en 96 hilos conductores separados, usados para control, direccionamiento y datos. Como todos los componentes del sistema comparten un conjunto de caminos, su uso debe estar controlado por la CPU. Esta arquitectura es altamente flexible, permitiendo conectar módulos al bus para crear varias configuraciones.

1970- Últimas generaciones.

Con el gran avance de la tecnología, la rápida introducción de nuevos productos y la importancia del software y las comunicaciones, así como del hardware, la clasificación en generaciones se vuelve cada vez menos clara y menos significativa. Se podría decir que la aplicación comercial de nuevos desarrollos resultó uno de los principales cambios de principios de los años 70', y los resultados de estos cambios duran todavía. Mencionaremos dos de los más importantes:

Memoria semiconductora

Fue creada con la intención de poder recuperar datos, ya que hasta 1970 se utilizaban anillos de material ferro-magnético para memoria, que era cara, voluminosa y usaba lectura destructiva. La memoria semiconductora era no destructiva, del tamaño de un solo núcleo de ferrita (uno de los componentes de las memorias anteriores), más barata y veloz y podía tener 256bits de memoria.

A partir de 1970, la memoria semiconductora ha tenido 8 generaciones: 1K, 4K, 16K, 64K, 256K, 1M, 4M y ahora 16M bits en un solo chip. Cada generación ha proporcionado cuatro veces más densidad de almacenamiento que la generación previa, junto con un menor costo por bit y una mayor velocidad de acceso.

Microprocesadores

La densidad de elementos de procesamiento también crece. Conforme el tiempo pasaba, en cada chip había más y más elementos, así que cada vez se necesitaban menos chips para construir un procesador.

En 1971 Intel desarrolló el 4004, el cual fue el primer chip que contenía todos los componentes de la CPU en un solo chip: el microprocesador había nacido.

El 4004 podía sumar dos números de 4 bits y multiplicar sólo con sumas sucesivas. Hoy en día parece de la prehistoria, pero marcó el comienzo de la evolución continua en capacidad y potencia de los microprocesadores, así que hay que tenerle respeto. En 1972 Intel lanzó el 8008, el primer microprocesador de 8 bits, casi dos veces más complejo que el 4004. En 1974 apareció el 8080, primer microprocesador de uso general de 8 bits, más rápido, con un conjunto de instrucciones más rico y capacidad de direccionamiento mayor que los anteriores.

Hacia el final de los 70' crearon los microprocesadores de propósito general de 16 bits. Uno de ellos fue el 8086. En 1981 Bell y HP desarrollaron microprocesadores de un solo chip de 32 bits. Intel introdujo el suyo, el 80386 en 1985.

1.5 Diseño para conseguir mejores prestaciones.

Desde la perspectiva de la organización y arquitectura de las computadoras, los bloques básicos de los potentes computadores de hoy en día son prácticamente los mismos que los del computador IAS de hace casi 50 años, mientras que, por otra parte, las técnicas para sacar hasta la última gota del rendimiento de los elementos disponibles se han vuelto cada vez más sofisticadas.

Velocidad del microprocesador

Mientras que los fabricantes de chips han estado ocupados aprendiendo cómo se fabrican chips de densidad cada vez mayor, los diseñadores del procesador tienen que producir técnicas cada vez más elaboradas para alimentar al *monstruo*. Entre las técnicas incorporadas a los procesadores de hoy en día están:

- **Predicción de ramificación:** el procesador se anticipa al software y predice qué ramas, o grupos de instrucciones, se van a procesar después con mayor probabilidad. Si el procesador acierta la mayoría de las veces, puede precaptar las instrucciones correctas y almacenarlas para mantener al procesador ocupado.
- **Análisis del flujo de datos:** el procesador analiza qué instrucciones dependen de los resultados de otras instrucciones o datos, para crear una organización optimizada de instrucciones. De hecho, las instrucciones se regulan para ser ejecutadas cuando estén listas, independientemente del orden original del programa. Esto evita retrasos innecesarios.
- **Ejecución especulativa:** utilizando los dos anteriores, algunos procesadores ejecutan especulativamente instrucciones antes de que aparezcan en la ejecución del programa, manteniendo los resultados en posiciones temporales. Esto

permite al procesador mantener sus máquinas de ejecución tan ocupadas como sea posible, ejecutando instrucciones que es probable que se necesiten.

Equilibrio de prestaciones

Es necesario ajustar la organización y la arquitectura de las prestaciones para compensar las desigualdades de capacidad entre los distintos componentes.

Mientras que la velocidad del procesador y la capacidad de la memoria han crecido rápidamente, la velocidad con la que los datos pueden ser transferidos entre la memoria principal y el procesador se ha quedado dramáticamente retrasada. La interfaz entre el procesador y la memoria principal es el camino más importante de todo el computador, ya que es el responsable de llevar el constante flujo de instrucciones y datos entre los chips de la memoria y el procesador. Si la memoria o la interfaz no logran mantener el ritmo de las insistentes demandas del procesador, éste se estanca en una posición de espera y se pierde así tiempo de procesamiento valioso.

Otra área de diseño se centra en el manejo de dispositivos de E/S. Conforme los computadores se hacen más rápidos y potentes, se desarrollan aplicaciones más sofisticadas, que se apoyan en el uso de periféricos con demandas intensivas de E/S. La generación actual de procesadores puede manejar los datos producidos por estos dispositivos, pero aún queda el problema de mover esos datos entre el procesador y los periféricos. Las estrategias en relación con esto incluyen esquemas de caches y almacenamiento, más el uso de buses de interconexión de más alta velocidad y con estructuras más elaboradas. Además, el uso de configuraciones multiprocesador puede ayudar a satisfacer las demandas de E/S.

1.6 Evolución del Pentium (CISC) y del Power PC (RISC).

Intel

- **8080:** primer microprocesador de propósito general del mundo. Era una máquina de 8 bits, con datos de memoria de 8 bits.
- **8086:** una máquina de 16 bits, más potente. Además de un camino de datos más ancho y registros más grandes, tenía una cache de instrucción, o cola, que precaptaba algunas instrucciones antes de ser ejecutadas.
- **80286:** una ampliación de la 8086, permitía direccional una memoria de 16Mb en lugar de sólo 1Mb.
- **80386:** la primera con 32 bits.
- **80486:** introduce el uso de cache más sofisticada y potente, e instrucciones de segmentación de cauce sofisticadas.
- **Pentium:** introduce el uso de técnicas superescalares, que permiten que varias instrucciones se ejecuten en paralelo.
- **Pentium Pro:** se continuó la tendencia de Pentium hacia la organización superescalar, con el uso agresivo del renombrado de registros, predicción de ramificaciones, análisis del flujo de datos y ejecución especulativa.
- **Pentium II:** se incorporó la tecnología Intel MMX, que se diseñó específicamente para procesar de forma eficiente datos de video, audio y gráficos.
- **Pentium III:** incorpora instrucciones adicionales en punto flotante para procesar software en formato 3D.
- **Merced:** usa una organización de 64 bits.

Power PC

- **601:** una máquina de 32 bits.
- **603:** igual a la 601 pero con costo más bajo e implementación más eficiente (para PC portátiles).
- **604:** es una máquina de 32 bits con técnicas de diseño superescalares, para lograr mayores prestaciones (para computadores de mesa y servidores).
- **620:** implementa una arquitectura completa de 64 bits, incluyendo registros y buses de datos de 64 bits (para servidores finales).
- **740/750 ó G2:** integra dos niveles de cache en el chip del procesador principal, ofreciendo mejores significativas en las prestaciones sobre otras máquinas comparables con organización de cache fuera del chip.
- **G4:** continuará incrementando el paralelismo y la velocidad interna del chip del procesador.

1.7 Definiciones varias

Fuerzas tecnológicas y económicas

La rapidez del progreso tecnológico puede modelarse de acuerdo con la *Ley de Moore*, que expresa que el número de transistores se duplica cada 18 meses.

Benchmark

El **benchmark** es una técnica utilizada para medir el rendimiento de un sistema o componente del mismo, frecuentemente en comparación con el que se refiere específicamente a la acción de ejecutar un benchmark. La palabra *benchmark* es un anglicismo traducible al castellano como *comparativa*. Si bien también puede encontrarse esta palabra haciendo referencia al significado original en la lengua anglosajona, es en el campo informático donde su uso está más ampliamente extendido. Más formalmente puede entenderse que un benchmark es el resultado de la ejecución de un programa informático o un conjunto de programas en una máquina, con el objetivo de estimar el rendimiento de un elemento concreto, y poder comparar los resultados con máquinas similares. En términos de ordenadores, un benchmark podría ser realizado en cualquiera de sus componentes, ya sea CPU, RAM, tarjeta gráfica, etc. También puede ser dirigido específicamente a una función dentro de un componente, por ejemplo, la unidad de coma flotante de la CPU; o incluso a otros programas.

La tarea de ejecutar un benchmark originalmente se reducía a estimar el tiempo de proceso que lleva la ejecución de un programa (medida por lo general en miles o millones de operaciones por segundo). Con el correr del tiempo, la mejora en los compiladores y la gran variedad de arquitecturas y situaciones existentes convirtieron a esta técnica en toda una especialidad. La elección de las condiciones bajo la cual dos sistemas distintos pueden compararse entre sí es especialmente ardua, y la publicación de los resultados suele ser objeto de cándentes debates cuando éstos se abren a la comunidad.

También puede realizarse un "benchmark de software", es decir comparar el rendimiento de un software contra otro o de parte del mismo, por ejemplo, comparar distintas consultas a una base de datos para saber cuál es la más rápida o directamente partes de código.

El Benchmark es también un proceso continuo de medir productos, servicios y prácticas contra competidores más duros o aquellas compañías reconocidas como líderes en la industria.

MIPS

Es el acrónimo de "millones de instrucciones por segundo". Es una forma de medir la potencia de los procesadores. Sin embargo, esta medida sólo es útil para comparar procesadores con el mismo juego de instrucciones y usando *benchmarks* que

fueron compilados por el mismo compilador y con el mismo nivel de optimización. Esto es debido a que la misma tarea puede necesitar un número de instrucciones diferentes si los juegos de instrucciones también lo son; y por motivos similares en las otras dos situaciones descritas.

Se define MIPS = F/ CPI, donde F es la frecuencia de reloj en MHz y CPI es la media de los ciclos de reloj que usan las instrucciones para ser ejecutadas. En el mundo de Linux se suelen referir a los MIPS como 'BogoMips'. El equivalente en la aritmética de punto flotante de los MIPS son los flops.

FLOPS

Es el acrónimo de *Floating point Operations Per Second* (operaciones de punto flotante por segundo). Se usa como una medida del rendimiento de una computadora, especialmente en cálculos científicos que requieren un gran uso de operaciones de coma flotante.

Las computadoras exhiben un amplio rango de rendimientos en punto flotante, por lo que a menudo se usan unidades mayores que el FLOPS. Los prefijos estándar del Sistema Internacional de Medidas pueden ser usados para este propósito, dando como resultado

1. **megaFLOPS** (MFLOPS, 10^6 FLOPS),
2. **gigaFLOPS** (GFLOPS, 10^9 FLOPS),
3. **teraFLOPS** (TFLOPS, 10^{12} FLOPS),
4. **petaFLOPS** (PFLOPS, 10^{15} FLOPS),
5. **exaFLOPS** (EFLOPS, 10^{18} FLOPS).

1.7 Niveles de abstracción

Para describir un sistema jerárquico se pueden utilizar dos aproximaciones: o bien se empieza por la parte más baja y se van construyendo los niveles de jerarquía basándose en el nivel inferior (modelo *bottom-up*), o bien se realiza una descripción desde el nivel más alto y se descompone el sistema en módulos de jerarquía inferior (modelo *top-down*). Una posible clasificación de los niveles estructurales, de menor a mayor, sería:

1. **Nivel de componente:** se trata con las leyes de la física de estado sólido, y la electrónica física. Los elementos de este nivel son difusiones de impurezas de tipo P y N en silicio, posilicio cristalino y difusiones de metal que sirven para construir los transistores.
2. **Nivel electrónico:** los componentes son transistores, resistencias, condensadores y diodos construidos con las difusiones del nivel anterior. Esta tecnología es la que se utiliza para fabricar circuitos integrados. En este nivel se construyen las puertas lógicas.
3. **Nivel digital:** sus elementos son las puertas lógicas y los biestables. Se aplica el álgebra booleana y la lógica digital.
4. **Nivel RTL:** nivel de transferencia de registros. Sus elementos son los registros y módulos combinacionales aritméticos.
5. **Nivel PMS:** sus siglas provienen del inglés *Processor Memory Switch* y sus elementos son los buses, las memorias, procesadores y otros módulos de alto nivel.

2. Aritmética de las computadoras

Definición de bit, nibble, byte, palabra, palabra doble, relación con lenguajes de alto nivel. Representaciones numéricas: números enteros con y sin signo. Aritmética con enteros. Fundamentos de la representación en punto flotante, normalización, error de la representación. Representación estándar del IEEE. Aritmética en punto flotante. Representaciones alfanuméricas, ASCII, EBCDIC.

2.1 Definición de bit, nibble, byte, palabra, relación con lenguajes de alto nivel

Bit es el acrónimo de *Binary digit* (dígito binario). Un bit es un dígito del sistema de numeración binario donde se usan dos dígitos: el cero (0) y el uno (1). El bit es la unidad mínima de información empleada en informática o en cualquier dispositivo digital. Con él, podemos representar dos valores cuales quiera, basta con asignar uno de esos valores al estado de "apagado" (0), y el otro al estado de "encendido" (1).

Nibble o cuado es el conjunto de cuatro bits. Es importante ya que puede representar una cifra en hexadecimal.

Byte es un conjunto de 8 bits contiguos. No es lo mismo que un octeto, el cual es un conjunto de 8 bits (dos nibbles). Un byte comprende el *sub-campo direccionable más pequeño* del tamaño de palabra natural de la computadora. Esto es, la unidad de datos binarios más pequeña en que la computación es significativa, o se pueden aplicar las cotas de datos naturales.

Palabra es una cadena finita de bits que son manejados como un conjunto por la máquina. El tamaño o longitud de una palabra hace referencia al número de bits contenidos en ella. El tamaño de una palabra se refleja en muchos aspectos de la estructura y las operaciones de las computadoras. La mayoría de los registros en un ordenador normalmente tienen el tamaño de la palabra. El valor numérico típico manipulado por un ordenador es probablemente el tamaño de palabra. La cantidad de datos transferidos entre la CPU del ordenador y el sistema de memoria a menudo es más de una palabra. Una dirección utilizada para designar una localización de memoria a menudo ocupa una palabra. Los ordenadores modernos normalmente tienen un tamaño de palabra de 16, 32 -Palabra doble- ó 64 bits.

La relación de estas definiciones con los lenguajes de alto nivel está dada por la elección de los tipos de datos de cada lenguaje, lo que determinará el rango en que puede manejarse cada tipo de dato. Por ejemplo, en C+ se elige un tipo de entero sin signo de 8 bits, el cual está restringido en el rango de 0 a 255.

2.2 Teorema Fundamental de la Numeración:

Se trata de un teorema que relaciona una cantidad expresada en cualquier sistema de numeración posicional con la misma cantidad expresada en el sistema decimal. El Teorema Fundamental de la Numeración (TFN), establece que en cualquier sistema de numeración posicional todos los números pueden expresarse mediante la siguiente fórmula:

$$N = \sum_{i=-k}^n d_i \cdot b^{i-k}$$

d_i = dígito

i = posición del dígito con respecto a la coma fraccionaria

N , número válido en el sistema de numeración.

b , base del sistema de numeración. Número de símbolos permitidos en el sistema.

d_i , un símbolo cualquiera de los permitidos en el sistema de numeración.

n ,: número de dígitos de la parte entera.

, coma fraccionaria. Símbolo utilizado para separar la parte entera de un número de su parte fraccionaria.

k ,: número de dígitos de la parte decimal.

La fórmula general para construir un número N , con un número finito de decimales, en un sistema de numeración posicional de base b es la siguiente:

$$N = \begin{cases} \langle d_{(n-1)} \dots d_1 d_0, d_{-1} \dots d_{-k} \rangle = \sum_{i=-k}^n d_i b^i \\ N = d_n b^n + \dots + d_1 b^1 + d_0 b^0 + d_{-1} b^{-1} + \dots + d_{-k} b^{-k} \end{cases}$$

El valor total del número será la suma de cada dígito multiplicado por la potencia de la base correspondiente a la posición que ocupa en el número.

Esta representación posibilita la realización de sencillos algoritmos para la ejecución de operaciones aritméticas.

2.3 Sistema Octal

El sistema numérico en base 8 se llama **octal** y utiliza los dígitos 0 a 7.

Por ejemplo, el número 74 (en decimal) es 1001010 (en binario), lo agruparíamos como 1 / 001 / 010, de tal forma que obtengamos una serie de números en binario de 3 dígitos cada uno (para fragmentar el número se comienza desde el primero por la derecha y se parte de 3 en 3), después obtenemos el número en decimal de cada uno de los números en binario obtenidos: 1=1, 001=1 y 010=2. De modo que el número decimal 74 en octal es 112.

2.4 Representaciones numéricas: números enteros con y sin signo

Números enteros sin signo (BSS)

Si el número tiene n bits se pueden representar 2^n números distintos. Por ejemplo: $2_{(10)} = 010_{(2)}$. El rango va desde 0 a (2^{n-1}) .

Números enteros con signo (BCS)

Con n bits, 1 bit representa al signo (el más significativo) y n-1 bits a la magnitud. Un 0 en el bit de signo indica que el número es positivo, un 1 que es negativo. Los otros bits representan el número. Por ejemplo: $+32_{(10)} = 00100000_{(2)}$. El rango va de $-(2^{n-1} - 1)$ a $+(2^{n-1} - 1)$, con dos representaciones del cero.

Números en punto fijo

Se considera que todos los números a representar tienen exactamente la misma cantidad de dígitos y la coma fraccionaria está siempre ubicada en el mismo lugar, como en el sistema decimal. En el sistema binario, por ejemplo, $11,10_{(2)} = 3,5_{(10)}$. La diferencia principal entre la representación en papel y su almacenamiento en la PC es que no se guarda coma alguna, se supone que está en un lugar determinado.

El rango es la diferencia entre el número mayor y menor, por ejemplo si n=3, el rango es $111-000=111$. La resolución es la diferencia entre dos números consecutivos.

Existen otras representaciones numéricas de enteros, que son más usadas para aritmética:

Técnica de Complementos

El complemento a un número N de un número A ($A < N$) es igual a la cantidad que le falta a A para ser N.

- Complemento a N de A = $N - A$
- El complemento a un número N del número $(N - A)$ es igual a A.
- Complemento a N de $(N - A) = N - (N - A) = A$

En un sistema con n dígitos podemos tener:

- Complemento a la base disminuida: si $N = \text{base}^n - 1$, es Ca1.
- Complemento a la base: si $N = \text{base}^n$ es complemento a 2, Ca2.

Complemento a 1

los n bits representan al número. Si el número es positivo, la representación es la misma. En cambio, si es negativo (comienza con 1), los n bits tienen el Ca1 del valor deseado. El Ca1 de un número en base 2 se obtiene invirtiendo todos los bits. Por ejemplo: $+32_{(10)} = 00100000$; $-32_{(10)} = 11011111$. El rango es desde $-(2^{n-1} - 1)$ a $+(2^{n-1} - 1)$, con dos ceros.

Nota

Si se le hace XOR a un número, se obtiene el Ca1 de ese número.

Complemento a 2

Los n bits representan al número. Si el número es positivo, los n bits tienen la representación binaria del número (como siempre). Si es negativo, los n bits tienen el Ca2 del valor deseado. El Ca2 de un número se obtiene con Ca1 y luego sumándole 1. Por ejemplo: $+32_{(10)} = 00100000_{(2)}$; $-32_{(10)} = 11100000_{(2)}$. El rango va desde $-(2^{n-1})$ a $+(2^{n-1} - 1)$, con un cero.

Técnica del exceso

La representación de un número A es la que corresponde a la suma del mismo y un valor constante E (o exceso). Dado un valor, el número representado se obtiene restando el valor del exceso. $A = (\text{exceso } E \text{ de } A) - E$. El signo del número A resulta de esta resta. Por ejemplo: si $n = 6$, exceso 32 = $-2^{6-1} = 000000_{(2)} = 0 - 32$. El rango va desde -2^{n-1} a $2^{n-1} - 1$.

Flags

Son bits que el procesador establece de modo automático acorde al resultado de cada operación realizada. Sus valores permitirán tomar decisiones, como realizar o no una transferencia de control o determinar relaciones entre números (mayor, menor, igual). Algunos tipos de banderas son

- **Z (cero):** vale 1 sólo si el resultado de la operación son todos bits 0.
- **C (carry):** vale 1 si en la suma hay acarreo. En la resta vale 1 si hay *borrow*. Cuando la operación involucra BSS, C en 1 indica que está fuera de rango.
- **N (negativo):** vale 1 si el resultado es negativo.
- **V (overflow):** en 1 indica una condición de fuera de rango en Ca2.

2.5 Aritmética con enteros

Negación en Ca2: se realiza siguiendo las siguientes reglas: 1. Obtener el complemento booleano de cada bit del entero. 2. Tratando el resultado como un entero binario sin signo, sumarle 1.

Suma en Ca2: se suman los bits directamente. Si se suman dos números positivos y el resultado es negativo, o se suman dos negativos y el resultado es positivo, existe *overflow*.

Resta en Ca2: se restan los bits directamente y sucede lo mismo que en la suma: si a un número positivo se le resta uno negativo y el resultado es negativo o, si a un número negativo se le resta un positivo y el resultado es positivo, existe *overflow*.

2.6 Fundamentos de la representación en punto flotante, normalización, error de la representación.

Punto fijo: error en punto fijo

El error absoluto máximo cometido puede considerarse como la mitad de la resolución (diferencia entre dos números consecutivos).

Números en punto flotante

Un número muy grande se puede representar, tanto en la aritmética decimal, como en la binaria de la forma $\pm M \times B^{\pm E}$, donde M es la mantisa, B es la base y E es el exponente.

Como la base en binario siempre es 2, es implícita y no necesita almacenarse. Entonces, almacenamos M y E en vez del número completo, ahorrando bits.

Mantisa y Exponente se almacenan en sistemas de punto fijo (BSS, BCS, Ca1, Ca2, Exceso) y de esto dependerá el rango y la resolución (el rango en punto flotante es más grande que en punto fijo).

Ejemplo: mantisa en Ca2 de 4 bits entera y exponente en Ca2 de 4 bits entero:

- Máximo : $0111 \times 2^{0111} = +7 \times 2^{+7}$
- Mínimo: $1000 \times 2^{0111} = -8 \times 2^{+7}$
- Rango: $[-8 \times 2^{+7} \dots +7 \times 2^{+7}]$
- Resolución en el extremo superior: $(7-6) \times 2^7 = 2^7$
- Resolución en el origen: $(1 \times 2^{-8} - 0) = 2^{-8}$

El formato para la representación en punto flotante es



Normalización

La **normalización** sirve para hacer operaciones aritméticas teniendo un único par de valores de mantisa y exponentes para un número. Para normalizar, las mantisas fraccionarias se definen de la forma:

$0,1\overline{ddd...dd}d$, donde d es un dígito binario (0 o 1).

Todas las mantisas comienzan con 0,1, por lo tanto el 1 no lo almacenamos (para ahorrar bits), entonces tenemos un bit más para representar. El 1 que no almacenamos se llama Bit Implícito y no hay que olvidarse que existe.

Ejemplo: mantisa BCS de 23 bits, fraccionaria normalizada y exponente Exc 8 bits entero.

- Max + = 0,0111111111111111111111111 $\times 2^{11111111} = +(1-2^{-23}) \times 2^{+127}$
- Min + ($\neq 0$) = 0,0100000000000000000000000 $\times 2^{00000000} = +(0,5) \times 2^{-128}$.
- Max - ($\neq 0$) = 1,0100000000000000000000000 $\times 2^{00000000} = -(0,5) \times 2^{-128}$
- Min - = 1,01111111111111111111111 $\times 2^{11111111} = -(1-2^{-23}) \times 2^{+127}$

Ahora, ¿Cómo se escribe un número en Punto Flotante normalizado?

1. Se escribe el número en el sistema propuesto para la mantisa.
2. Se desplaza la coma y se cambia el exponente hasta obtener la forma normalizada (se le suma tanto como lugares se corrió la coma hacia la izquierda).
3. Se convierte el exponente al sistema propuesto para él.

Ejemplo: $-13,5_{(10)}$ en el formato del ejemplo anterior

1. $1^{\circ} 1101,1000 \dots 00000 = 1,1101,100 \dots 000 \times 2^0$
2. $1,01101100 \dots 00000 \times 2^4$ (se corre la coma 4 lugares, se suma 4 al exponente).
3. 4 en Ca2 = 00000100; 4 en exceso = 10000100.

Y, finalmente, lo representamos en la recta:

- Sin bit implícito: 1 10000100 1101100000000000000000000
- Con bit implícito: 1 100001000 1011000000000000000000000

Resolución

Es la diferencia entre dos representaciones sucesivas y varía a lo largo del rango.
No es constante como en el caso de punto fijo.

Error absoluto

Es la diferencia entre el valor representado y el valor a representar. Error absoluto máximo \leq resolución/2

Error relativo

Es el error absoluto dividido por el número a representar (expresado en porcentaje).

2.7 Representación estándar del IEEE

Estándar IEEE 754

Se desarrolló para facilitar la portabilidad de los programas de un procesador a otro y para alentar el desarrollo de programas numéricos sofisticados.

Existen dos representaciones: simple precisión y doble precisión. La mantisa es fraccionaria normalizada, con la coma después del primer bit (que es siempre 1).

El exponente está representado en exceso ($2^{n-1} - 1$).

IEEE 754	Simple Precisión	Doble Precisión
Bits en signo	1	1
Bits en exponente	8	11
Bits en fracción	23	52
Total de bits	32	64
Exponente en exceso	127	1023
Rango de exponente	-126 a +127	-1022 a +1023
Rango de números	2^{-126} a 2^{128}	2^{-1022} a 2^{1024}

Caso especial: simple precisión

Exponente	Mantisa	Valor
00000000<E<11111111	Cualquiera	$\pm 1, M \times 2^{(E_{\text{ess}})-127}$
11111111	$\neq 0$	NAN
11111111	=0	\pm Infinito
00000000	$\neq 0$	$\pm 0, M \times 2^{-126}$
00000000	=0	± 0

Caso especial: doble precisión

Exponente	Mantisa	Valor
000000000000<E<111111111111	Cualquiera	$\pm 1, M \times 2^{(E_{\text{ess}})-1023}$
111111111111	$\neq 0$	NAN
111111111111	=0	\pm Infinito
000000000000	$\neq 0$	$\pm 0, M \times 2^{-1022}$
000000000000	=0	± 0

2.8 Operaciones aritméticas en punto flotante

Suma y Resta.

Cuando sumamos o restamos dos números en coma flotante se deben comparar los exponentes y hacerlos iguales; para lo cual hay que desplazar o alinear uno de ellos respecto al otro. Dados dos números en representación en coma flotante como

$$x = m_x \cdot 2^{x_e} \quad y = m_y \cdot 2^{y_e}$$

las operaciones de suma y resta se definen de la siguiente forma, suponiendo que $x_e < y_e$:

$$x + y = (m_x \cdot 2^{x_e} + m_y \cdot 2^{y_e}) \cdot 2^{y_e}$$

$$x - y = (m_x \cdot 2^{x_e} - m_y \cdot 2^{y_e}) \cdot 2^{y_e}$$

Las operaciones de suma y resta, así como la multiplicación y la división pueden producir desbordes, por producir resultados demasiado grandes (desbordamientos) o demasiado pequeños (subdesbordamientos). Hay cuatro tipos de errores posibles:

1. **Desbordamiento del exponente.** Es cuando un exponente positivo E excede de su valor máximo permitido. En algunos ordenadores el número X se representa entonces como + o -.
2. **Subdesbordamiento del exponente.** Es cuando un exponente negativo E excede de su valor mínimo permitido. Esto significa que el numero X es demasiado pequeño y se puede considerar como igual a 0.

3. **Subdesbordamiento de mantisa.** En el proceso de alineación de las mantisas, si los dígitos se desplazan hacia la derecha más allá de su bit menos significativo, lo que sucede es que se pierden y es como redondear el resultado.
4. **Desbordamiento de mantisa.** En la suma de dos mantisas del mismo signo se puede producir un arrastre del bit más significativo. Esto se soluciona mediante la re-normalización, desplazando a la derecha un bit la mantisa y ajustando el exponente.

Algoritmo de la suma-resta.

Por lo visto con anterioridad, para realizar la suma (resta) de dos operandos en representación coma flotante debemos realizar previamente la separación de los exponentes y de las mantisas para su tratamiento posterior, y después realizar esta serie de pasos:

1. Seleccionar el número con menor exponente y desplazar su mantisa a la derecha tantas veces como indique la diferencia en módulo de los exponentes.
2. Hacer que el exponente resultado sea igual al mayor de los exponentes.
3. Realización de las operaciones de suma o resta con las mantisas.
4. Normalización del resultado. Una vez realizada la suma se debe normalizar desplazando los bits de la mantisa hacia la izquierda o la derecha con lo cual habrá que cambiar el valor del exponente.
5. Comprobar las condiciones de rebote.

Multiplicación

La multiplicación y la división en punto flotante son más sencillas de realizar. Vemos la fórmula que nos permitirá realizar estas operaciones manualmente:

$$x * y = (m_x * m_y) 2^{e_x + e_y}$$

$$\frac{x}{y} = (m_x * m_y^{-1}) 2^{e_x - e_y}$$

Algoritmo de multiplicación y división.

Este algoritmo está fundamentado en el seguimiento de cuatro pasos:

1. Realizar la Suma-Resta de los exponentes.
2. Multiplicar-dividir las mantisas y determinar el signo del resultado.
3. Normalizar el valor resultado, si es necesario.
4. Comprobar las condiciones de rebote.

Detalles a tener en cuenta

Sumar y restar

- Comprobar valores cero.
- Ajustar mantisas (ajuste de exponentes).
- Sumar o restar las mantisas.
- Normalizar el resultado.

Multiplicar y dividir

- Comprobar valores cero
- Sumar y restar exponentes
- Multiplicar y dividir mantisas (tener en cuenta el signo)
- Normalizar
- Redondear

Todos los resultados intermedios deben doblar su longitud al almacenarse.

2.9 Representaciones alfanuméricas, ASCII, EBCDIC.

En representación alfanumérica se pueden representar

- Letras (mayúsculas y minúsculas).
- Dígitos decimales (0..9).
- Signos de puntuación (; , . : etc.).
- Caracteres especiales (* @ + =, etc.).
- Caracteres u órdenes de control (TAB – Intro, etc.).

A cada símbolo le corresponde un código en binario, que puede ser, entre otros:

- FIELDATA: 26 letras mayúsculas + 10 dígitos + 28 caracteres especiales.
 - Total= 64 combinaciones → código de 6 bits.
- ASCII: FIELDATA+minúsculas+ctrl.
 - Total= 128 combinaciones → código de 7 bits.
- ASCII extendido: ASCII+multinacional+semigráficos+matemática
 - Total= 256 combinaciones → código de 8 bits.
- EBCDIC (extended BCD Interchange Code): similar a ASCII pero de IBM. Código de 8 bits.

3. Lógica Digital

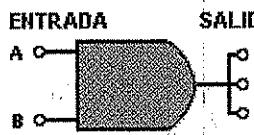
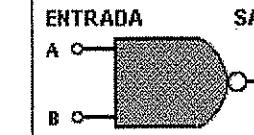
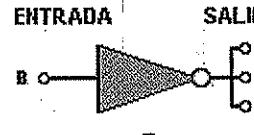
Compuertas lógicas. Álgebra de Boole. Implementación de funciones booleanas. Lógica combinatoria, codificadores, decodificadores, multiplexores. Lógica secuencial, registros, contadores. Concepto de memoria y lógica programable. Aplicaciones de lógica digital en la Unidad Aritmético - Lógica (ALU) y en la Unidad de Control.

3.1 Compuertas lógicas

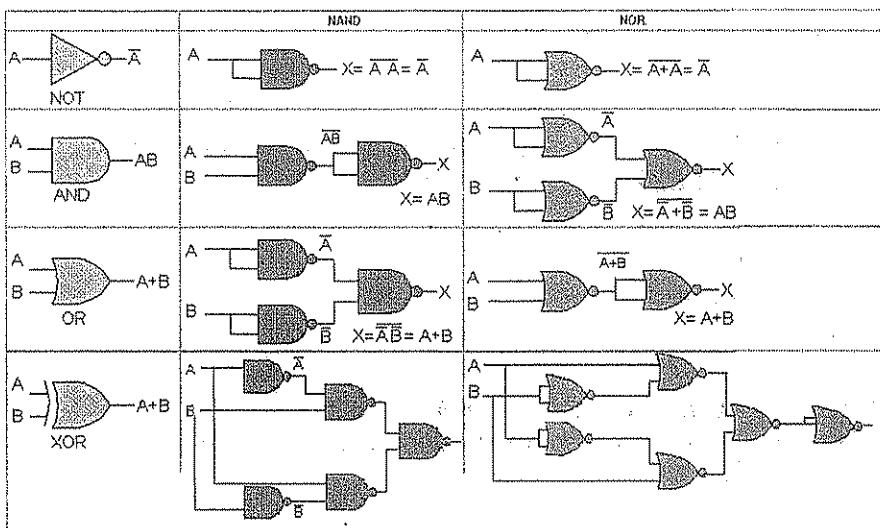
Son diminutos dispositivos electrónicos que pueden calcular diversas funciones con las señales 0 y 1. Las compuertas constituyen la base de hardware sobre la que se construyen todas las computadoras digitales.

Existen seis tipos de compuertas: NOT o inversora, AND, OR, XOR (OR exclusiva), NAND (NOT AND) y NOR (NOT OR), siendo las dos últimas las más importantes, llamadas “burbujas de inversión” o compuertas universales, ya que las otras se basan en ellas para su construcción.

Representación y equivalencias

PUERTA AND (A y B)  ENTRADA SALIDA $C = A \cdot B$	PUERTA NAND (no A y B)  ENTRADA SALIDA $C = \overline{A \cdot B}$
PUERTA OR (A o B, o ambos)  ENTRADA SALIDA $C = A + B$	PUERTA NOR (ni A ni B, ni ambos)  ENTRADA SALIDA $C = \overline{A + B}$
PUERTA NOT (no C)  ENTRADA SALIDA $C = \overline{B}$	PUERTA XOR (A o B, pero no ambos)  ENTRADA SALIDA $C = A \cdot \overline{B} + \overline{A} \cdot B$

1. Representación y tablas de verdad de las compuertas lógicas



2 Equivalencia de las compuertas lógicas

3.2 Álgebra de Boole

Se usa para describir los circuitos que pueden construirse combinando compuertas, donde las variables y funciones sólo pueden adoptar los valores 0 y 1.

Una función booleana tiene una o más variables de entrada y produce un resultado que depende sólo de los valores de dichas variables. Puesto que una función booleana de n variables sólo tiene 2^n posibles combinaciones de los valores de entrada, la función puede describirse totalmente con una *tabla de verdad* de 2^n renglones, cada uno de los cuales indica el valor de la función para una combinación distinta de valores de entrada.

Distintas combinaciones

A	B	NOT A	NOT B	A OR B	A AND B	A XOR B	A NOR B	A NAND B	A XNOR B
0	0	1	1	0	0	0	1	1	1
0	1	1	0	1	0	1	0	1	0
1	0	0	1	1	0	1	0	1	0
1	1	0	0	1	1	0	0	1	1

Nota

OR siempre es 1 cuando hay un uno en las variables.

AND siempre es 0 cuando hay un cero en las variables.

XOR es 0 cuando ambas son iguales y 1 cuando son distintas.

3.3 Implementación de funciones booleanas

Para implementar una función (F) booleana se usa el “método general para determinar una fórmula lógica dada una tabla de verdad: el método de la suma y el producto (SOP)”. Con ella, dada una tabla de verdad, se continúa con los siguientes dos sencillos pasos:

1. Verificar las salidas con valor 1.
2. Sumar todos los términos.

Así, se obtiene F y se puede hacer el dibujo de los circuitos, usando compuertas AND y OR, o cambiándolas por compuertas universales o a las otras.

Por ejemplo:

A	B	F	RESULTADO
0	0	0	A NAND B
0	1	1	NOT A AND B
1	0	0	NOT A NAND B
1	1	1	A AND B

Por lo tanto, F es igual a NOT A AND B OR A AND B.

Se puede obtener también otra forma de tabla de verdad, mediante el método del producto de sumas (POS), inversa a SOP y su lógica se basa en que, si la forma SOP indica que la salida es 1, si cualquiera de las combinaciones de entrada que producen 1 es cierta, entonces se puede decir que la salida es 1, si ninguna de las combinaciones de entrada que producen 0 también es cierta.

Por lo tanto, F es igual a A NAND B OR NOT A NAND B.

3.4 Lógica combinatoria, codificadores, decodificadores, multiplexores

Circuitos combinacionales

Un **circuito combinacional** es un conjunto de puertas interconectadas, cuya salida, en un momento dado, es función solamente de la entrada en ese instante. Se puede definir de tres formas:

1. **Tabla de verdad:** para cada una de las 2^n combinaciones posibles de las n señales de entrada, se enumera el valor binario de cada una de las m señales de salida.
2. **Símbolo gráfico:** describe la organización de las interconexiones entre puertas.
3. **Ecuaciones booleanas:** cada señal de salida se expresa como una función booleana de las señales de entrada.

Un circuito combinacional:

- Responde a los valores lógicos en las entradas, la salida está determinada exclusivamente por los valores de las entradas en ese instante.

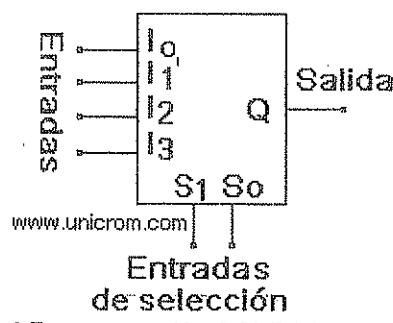
- Si cambia la entrada, cambia la salida.
- Los valores pasados de las entradas no influyen en los valores de las salidas.

Equivalencias de circuitos

Nombre	Función AND (*)	Función OR (+)
Ley de identidad	$1 * A = A$	$0 + A = A$
Ley nula	$0 * A = 0$	$1 + A = 1$
Ley de idempotencia	$A * A = A$	$A + A = A$
Ley inversa	$A * \text{NOT } A = 0$	$A + \text{NOT } A = 1$
Ley conmutativa	$A * B = B * A$	$A + B = B + A$
Ley asociativa	$(A * B) * C = A * (B * C)$	$(A + B) + C = A + (B + C)$
Ley distributiva	$A + B * C = (A+B)*(A+C)$	$A * (B + C) = A*B + A*C$
Ley de absorción	$A * (A + B) = A$	$A + A * B = A$
Ley de De Morgan	$\text{NOT}(A*B) = \text{NOT } A + \text{NOT } B$	$\text{NOT}(A+B) = \text{NOT } A * \text{NOT } B$

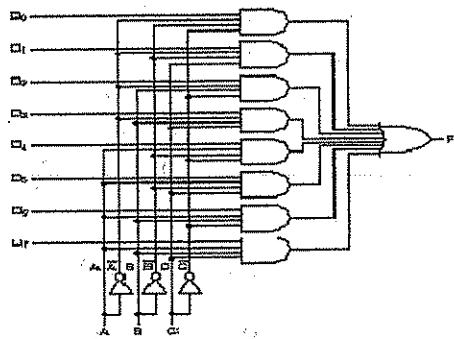
Multiplexores

El multiplexor conecta varias entradas con una única salida. En un momento dado, se selecciona una de las entradas para que pase a la salida. Los multiplexores se usan en circuitos digitales para controlar el enruteamiento de señales y datos. También se usan como convertidor de datos paralelos a serie.



3 Representación del Multiplexor

En el nivel de lógica digital, un multiplexor es un circuito con 2^n entradas de datos, una salida de datos y n entradas de control que seleccionan una de las entradas de datos. La entrada de datos seleccionada se “pasa por compuertas” (se encamina) hacia la salida. La siguiente figura es un diagrama esquemático de un multiplexor de 8 entradas. Las tres líneas de control, A, B y C, codifican un número de 3 bits que especifica cuál de las ocho líneas de entrada se encamina a la compuerta OR y de ahí a la salida. Sea cual sea el valor que esté en las líneas de control, siete de las compuertas AND siempre producirán 0; la otra podría producir 0 ó 1, dependiendo del valor de la línea de entrada seleccionada. Cada compuerta AND se habilita con una combinación distinta de las entradas de control.



Un ejemplo es la carga del contador de programa. El valor a cargar en el PC puede venir de una o varias fuentes diferentes:

- De un contador binario, si el PC se va a incrementar para la siguiente instrucción.
- Del registro de instrucción, si se acaba de ejecutar una instrucción de salto usando direccionamiento directo.
- De la salida de la ALU, si la instrucción del salto especifica la dirección usando modo de desplazamiento.

Las distintas entradas se pueden conectar a las líneas de entrada de un multiplexor con el PC conectado a la línea de salida. Las líneas seleccionadas determinan cuál es el valor a cargar en el PC. Como el PC tiene varios bits, se usan varios multiplexores, uno por bit.

Codificadores

Un **codificador** es un circuito combinacional con 2^n entradas y n salidas, cuya misión es presentar en la salida el código binario correspondiente a la entrada activada.

Existen dos tipos fundamentales de codificadores: codificadores sin prioridad y codificadores con prioridad. En el caso de codificadores sin prioridad, puede darse el caso de salidas cuya entrada no pueda ser conocida; por ejemplo, la salida 0 podría indicar que no hay ninguna entrada activada o que se ha activado la entrada número 0. Además, ciertas entradas pueden hacer que en la salida se presente la suma lógica de dichas entradas, ocasionando mayor confusión. Por ello, este tipo de codificadores es usado únicamente cuando el rango de datos de entrada está correctamente acotado y su funcionamiento garantizado.



4 Representación del Codificador Binario

Para evitar los problemas anteriormente comentados, se diseñan los codificadores con prioridad. En estos sistemas, cuando existe más de una señal activa, la

salida codifica la de mayor prioridad (generalmente correspondiente al valor decimal más alto). Adicionalmente, se codifican dos salidas más: una indica que ninguna entrada está activa, y la otra que alguna entrada está activa. Esta medida permite discernir entre los supuestos de que el circuito estuviera deshabilitado por la no activación de la señal de capacitación, que el circuito no tuviera ninguna entrada activa, o que la entrada número 0 estuviera activada.

Decodificadores

Un decodificador es un circuito combinacional con varias líneas de salida, con una sola de ellas seleccionada en un instante dado, dependiendo del patrón de líneas de entrada. En general, un decodificador tiene n entradas y 2^n salidas.

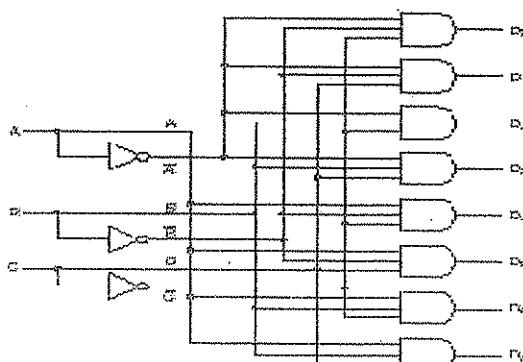
Los decodificadores tienen muchos usos en computadoras digitales. Su función principal es la de direccionar espacios de memoria. Un decodificador de n entradas puede direccionar 2^n espacios de memoria.

Para poder direccionar 1Kb de memoria necesitaría 10 bits, ya que la cantidad de salidas sería 2^{10} , igual a 1024.

De esta manera:

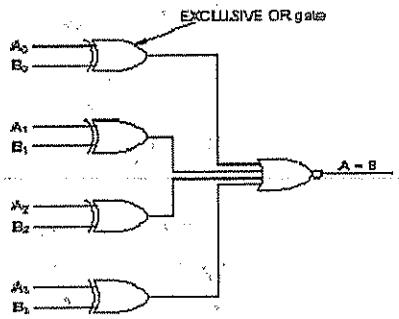
- Con 20 bits tengo 2^{20} que es 1Mb.
- Con 30 bits tengo 2^{30} que es 1Gb.

Con una línea adicional de entrada, se puede usar el decodificador como un desmultiplexor, el cual realiza la función inversa a un multiplexor. Para cada combinación de las entradas A, B y C sólo una de las salidas D_x vale '1'



Comparadores

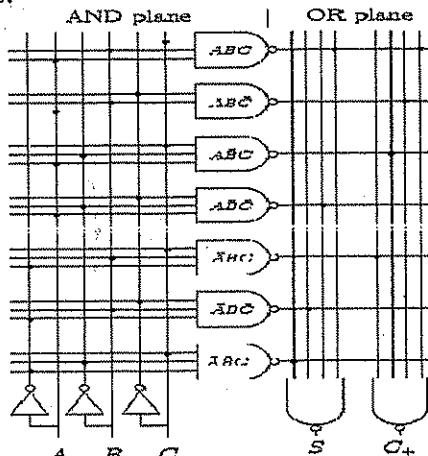
El comparador compara dos palabras de entrada. El comparador de la siguiente figura acepta dos entradas, A y B, cada una de 4 bits, y produce 1 si son iguales y 0 en caso contrario. El circuito se basa en la compuerta XOR. Si las dos palabras de entrada son iguales, las cuatro compuertas XOR deberán producir 0. Estas cuatro señales se conectan después a una compuerta OR; si el resultado es 0, las palabras de entrada son iguales; si es 1, no lo son. En la figura se invierte el sentido de la prueba con una NOR.



Arreglo Lógico Programable (PLA)

Se desarrolló con el objetivo de disminuir los problemas de diseño que tenía hacer chips lógicos con sus compuertas e interconexiones, que generaban grandes costos y mucho tiempo. El PLA es un chip de uso general que se adapta a usos específicos.

Los PLA se basan en el hecho de que cualquier función booleana se puede expresar en forma de SOP. Un PLA consiste en una disposición regular de puertas NOT, AND y OR en un chip. Cada entrada al chip pasa a través de una puerta NOT, así que cada entrada y su complemento están disponibles para cada puerta AND. La salida de cada puerta AND está disponible para cada puerta OR, y la salida de cada puerta OR es una salida del chip. Haciendo las conexiones adecuadas, se pueden implementar expresiones SOP arbitrarias.



Memoria de sólo lectura (ROM)

Como la información almacenada en la ROM se creó en el proceso de fabricación y es permanente, una entrada dada a la ROM (líneas de direcciones) siempre produce la misma salida (líneas de datos). Como las salidas son función sólo de las entradas presentes, la ROM es, de hecho, un circuito combinacional.

Se puede implementar una ROM con un decodificador y un conjunto de puertas OR.

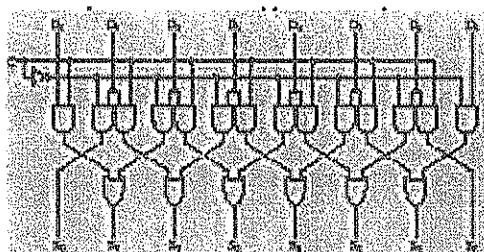
3.5 Lógica secuencial, registros, contadores

Circuitos aritméticos

Desplazadores

A	B	Carry IN	Suma	Carry Out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Según el valor de la entrada C los bits se ‘correrán’ un lugar a derecha o izquierda, perdiendo los bits que sobrepasan el rango de representación.



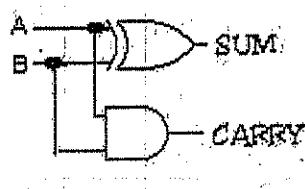
Para ver cómo funciona el circuito, observe los pares de compuertas AND para todos los bits excepto las compuertas del final. Cuando C=1, el miembro derecho de cada par se enciende, y pasa el bit de entrada correspondiente a la salida. Puesto que la compuerta AND derecha está conectada a la salida de la compuerta OR que está a su derecha, se efectúa un desplazamiento hacia la derecha. Cuando C=0, es el miembro izquierdo del par de compuertas AND el que se enciende, y se efectúa un desplazamiento a la izquierda.

Sumadores

Para poder sumar dos números de n bits se usan los sumadores.

Medio Sumador

Acepta dos dígitos binarios en las entradas y produce dos dígitos binarios en sus salidas, un bit con la suma y un bit para el carry. El carry de salida (Cout) es 1 solamente cuando A y B son 1, por lo tanto Cout se puede expresar como el AND de las variables de entrada. La salida con la suma es 1 solo cuando las variables de entrada A y B no son iguales, por lo tanto la suma se puede expresar como el XOR de las variables de entrada.

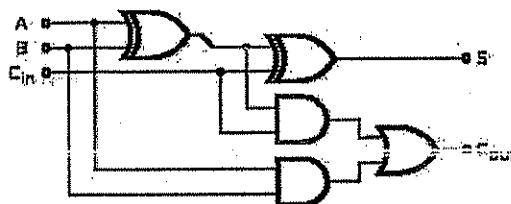


A	B	Suma	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Aunque un medio sumador es suficiente para sumar los bits de orden bajo de dos palabras de entrada de varios bits, no sirve para una posición de bit interior de la palabra porque no maneja el acarreo que llega de la posición que está a su derecha. Se necesita un sumador completo, el cual se construye de dos medios sumadores.

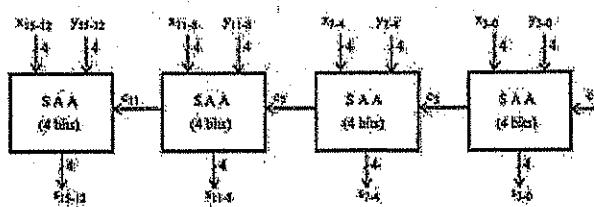
Sumador completo

Es un circuito combinacional que forma la suma aritmética de tres bits de entrada. Consta de tres entradas y dos salidas. Dos de las variables de entrada, que se indican por A y B, representan los dos bits significativo que van a añadirse. La tercera entrada, C, representa la cuenta que se lleva de la posición previa significativa más baja. Son necesarias dos salidas, debido a que la suma aritmética de tres dígitos binarios varía en valor de 0 a 3. Las dos salidas se denotan por los símbolos S para la suma y C para la cuenta que se lleva. La variable S da el valor del bit menos significativo de la suma. La variable binaria C da la cuenta que se lleva de salida. La tabla de verdad del sumador completo es como sigue



La línea de la salida Suma es 1 si el número de unos en A, B y el acarreo de entrada es impar. Acarreo de salida es 1 si A y B son ambos 1 o uno y sólo uno de ellos es 1 y el bit de acarreo de entrada es también 1. Juntos, los dos medios sumadores generan los bits tanto de suma como de acarreo.

Si queremos construir un sumador para, digamos, dos palabras de 16bits, basta con repetir el circuito anterior 16 veces. El acarreo de salida de un bit se usa como acarreo de entrada de su vecino izquierdo. El acarreo de entrada del bit de la extrema derecha se conecta a 0. Este tipo de sumador se conoce como sumador con propagación de acarreo.



5 Representación del Sumador

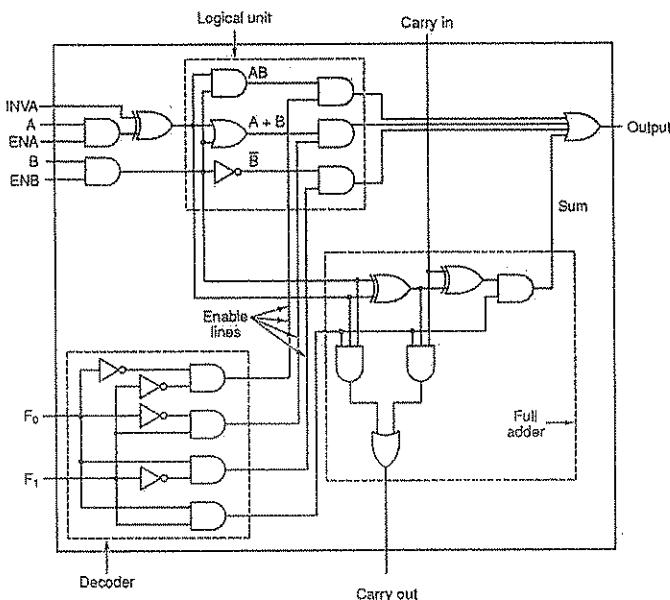
Hay que notar que, como la salida de cada sumador depende del acarreo del sumador previo, hay un retardo que crece del bit menos significativo al más significativo. Para sumadores grandes, este retardo puede hacerse inaceptablemente alto.

Si los valores del acarreo se pudieran determinar sin tener que pasar a través de todas las etapas previas, entonces este sumador de un bit podría funcionar independientemente, y el retardo no se acumularía. Éste se puede lograr con un procedimiento llamado *acarreo anticipado*. Cada término del acarreo se puede expresar en forma SOP como función sólo de las entradas originales, sin dependencia de los acarreos. Por lo tanto, sólo se dan dos niveles de retardo de puerta.

Para números grandes, este procedimiento se vuelve demasiado complicado, por consiguiente, el acarreo anticipado se hace normalmente con 4 u 8 bits a la vez.

Unidades aritmética lógica

Casi todas las computadoras contienen un solo circuito para obtener el AND, el OR y la suma de dos palabras de máquina. Por lo regular, un circuito de este tipo para palabras de n bits se construye con n circuitos idénticos para las posiciones de bits individuales. Un ejemplo sencillo de un circuito de este tipo es la ALU. Una ALU puede calcular cualquiera de cuatro funciones: A AND B, A OR B, NOT B, o A + B (suma aritmética), dependiendo de si las líneas de entrada de selección de función F0 y F1 contienen 00, 01, 10 o 11.



La esquina inferior izquierda de esta ALU de 1 bit contiene un decodificador de 2 bits que genera señales de habilitación para las cuatro operaciones, con base en las señales de control F0 y F1. Dependiendo de los valores de F0 y F1, una y sólo una de las cuatro líneas de habilitación se selecciona. Un valor de 1 en estas líneas permite que la salida de la función seleccionada pase hasta la compuerta OR final y de ahí a la salida.

La esquina superior izquierda contiene la lógica necesaria para calcular A AND B, A OR B Y NOT B, pero sólo uno de estos resultados pasa a la compuerta OR final,

dependiendo de las líneas de habilitación que salen del decodificador. Dado que una y sólo una de las líneas del decodificador es 1, una y sólo una de las cuatro compuertas AND que alimentan a la compuerta OR estará habilitada; las otras tres producirán 0, sean cuales sean los valores de A y B.

Además de poder usar A y B como entradas de operaciones aritméticas o lógicas, es posible hacer que cualquiera de las dos sea 0 negando ENA o ENB, respectivamente. También es posible obtener NOT A poniendo INVA. En condiciones normales, tanto ENA como ENB son 1 para habilitar ambas entradas, e INVA es 0. En este caso, A y B simplemente se alimentan a la unidad lógica sin modificación.

La esquina inferior derecha de la ALU contiene un sumador completo para calcular la suma de A y B, e incluye manejo de acarreos porque es probable que se conecten juntos varios de estos circuitos para efectuar operaciones con palabras enteras. Es posible conseguir circuitos como el de la siguiente figura, llamado *porción de bit (bit slice)*. Estas porciones permiten al diseñador de una computadora construir una ALU de la capacidad deseada. La señal INC sólo es útil en operaciones de suma; si está presente, incrementa en 1 el resultado, lo que permite calcular sumas como A + 1 y B + 1.

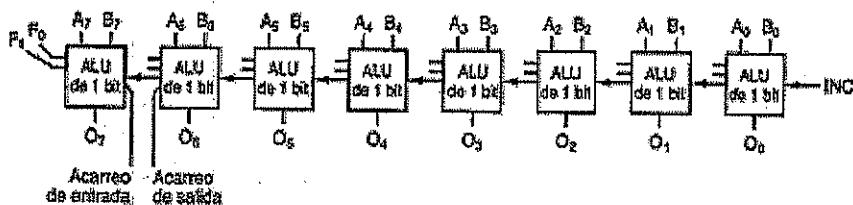


Figura 3-20. Ocho "tajadas" de ALU de un bit concordadas para formar una ALU de ocho bits. Por sencillez, no se muestran las señales de habilitación y de inversión.

Circuitos secuenciales

Se usan para proporcionar memoria o información de estado. La salida actual de un circuito secuencial depende no sólo de la entrada actual, sino también de la historia pasada de las entradas. La salida actual depende de la entrada actual y del estado actual del circuito.

Biestables

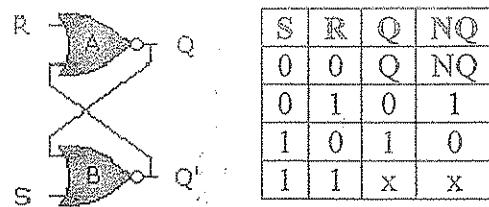
El biestable es un dispositivo con dos estados. Está en uno de los dos estados, en ausencia de entrada, recordando el último estado. Entonces, el biestable puede funcionar como una memoria de 1 bit. El biestable tiene dos salidas, que son siempre complementarias. Normalmente se denominan Q y NOT Q (NQ).

Tipos de biestables

1. Biestable S-R (latch S-R)

El circuito tiene dos entradas, S (Set) y R (Reset), y dos salidas, Q y NQ, y consiste en dos puertas NOR conectadas por realimentación. Este tipo de circuitos puede funcionar como una memoria de 1 bit, las entradas S y R sirven para escribir los

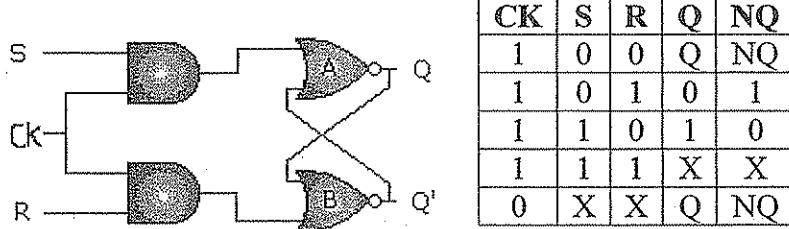
valores, 1 y 0 en la memoria. El cetrojo S-R es asincrónico porque, si las entradas no cambian, tampoco cambian las salidas.



Las combinaciones $S=R=1$ no son válidas, porque producirían una salida inconsistente ($Q=NQ=0$).

2. Biestable S-R síncrono

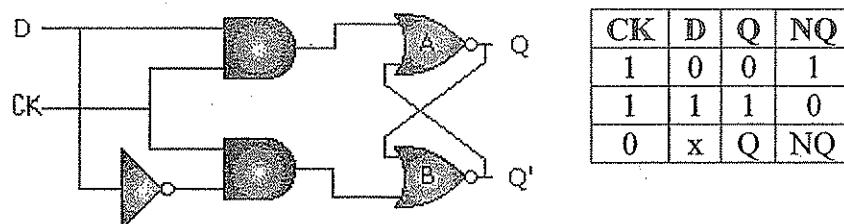
La diferencia con el S-R es que los cambios sólo ocurren en un pulso de reloj (entrada de 1 bit que va cambiando de 0 a 1 en tiempos regulares, constantemente).



3. Biestable D

Es un circuito de una sola entrada (D), la cual copia su valor a Q. Tiene la ventaja de que no hay error cuando las entradas son 1.

Como la salida del biestable D es siempre igual al valor más reciente aplicado a la entrada, recuerda y produce la última entrada. Por lo tanto sirve para guardar un bit. También se le llama “biestable de retardo” porque retrasa un 0 o un 1 aplicado a la entrada durante un pulso de reloj.



4. Biestable J-K

En base al biestable S-R, tiene dos entradas, pero en este caso todas las combinaciones posibles de los valores de entrada son válidos. Cuando $J=K=1$, la salida se invierte (función de conmutación).

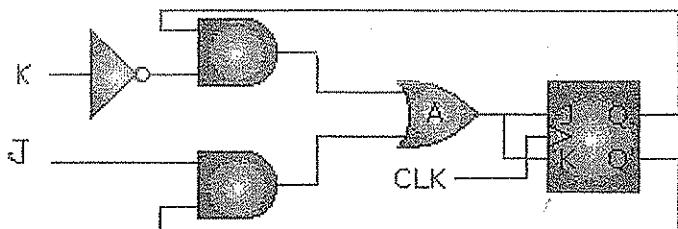


Diagrama de la catedra

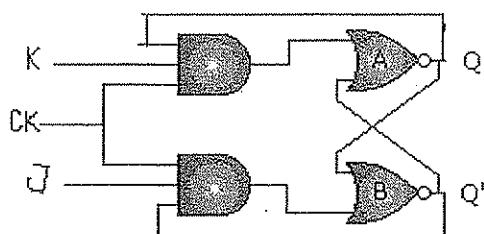
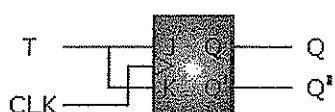


Diagrama de Stalling

CK	J	K	Q	NQ
1	0	0	Q	NQ
1	0	1	0	1
i	i	0	i	0
1	1	1	NQ	Q
0	X	X	Q	NQ

5. Biestable T

Se construye a partir de un J-K y siempre se niegan las salidas. Las dos entradas siempre valen 1 y lo que genera cambios en la salida es el reloj, por lo tanto se considera un temporizador.



T	Q	NQ
1	NQ	Q
X	Q	NQ

Registros

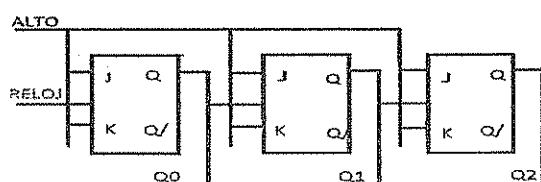
Un registro es un circuito digital usado en la CPU para almacenar uno o más bits de datos. Existen dos tipos:

1. **Registros paralelos:** consisten en un conjunto de memorias de 1 bit que se pueden leer o escribir simultáneamente. Para su funcionamiento, se usan biestables S-R. Una señal de control, llamada *validación de dato de entrada*, controla la escritura en los registros de los valores provenientes de las líneas de señales. Estas líneas pueden ser salidas de multiplexores, ya que los datos que se pueden cargar en un registro pueden provenir de una gran variedad de fuentes. La salida se controla de una forma similar. Como una característica extra, hay disponible una línea de puesta a cero (reset), que permite poner a 0 fácilmente el registro.
2. **Registros de desplazamiento:** un registro de este tipo acepta y/o transfiere información vía serie. Se pueden usar como interfaz de dispositivos serie de E/S. Además, pueden usarse en la ALU para realizar desplazamientos lógicos y funciones de rotación. En este último uso, necesitan equiparse con circuitería de lectura/escritura, tanto paralela como serie.

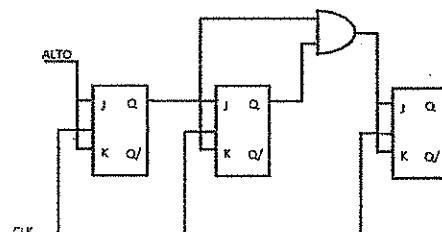
Contadores

Un contador es un registro cuyo valor se puede incrementar en 1 módulo la capacidad de ese registro. Así, un circuito hecho con n biestables puede contar hasta $2^n - 1$. Cuando el contador se incrementa más allá de su valor máximo, se pone a 0. Un ejemplo es el contador de programa.

Los contadores pueden ser síncronos o asíncronos. Los contadores asíncronos son relativamente lentos, ya que la salida de un biestable produce un cambio en el estado del siguiente biestable.



En un contador síncrono, todos los biestables cambian de estado a la vez. Como el último tipo es mucho más rápido, es el tipo que se usa en la CPU.



Si es importante más información sobre esto, consultar Apéndice A de Stallings.

3.6 Aplicaciones de lógica digital en la Unidad Aritmético - Lógica (ALU) y en la Unidad de Control.

UC. Lógica en la Unidad de Control

La lógica interna de la UC produce señales de control de salida a partir de las señales de entrada. La UC debe controlar el estado del ciclo de instrucción. Al final de cada subciclo, la UC emite una señal que hace que el generador de temporización (estructura entre el reloj y la UC) se reinicie y emita el primer tiempo. Además, la UC debe de establecer los valores adecuados de las señales de control para definir el siguiente subciclo a ejecutar.

ALU

Extraido de : http://es.wikipedia.org/wiki/Unidad_aritm%C3%A9tica_l%C3%B3gica

La ALU se compone básicamente de: circuito operacional, registros de entradas, un registro acumulador y un registro de estados, conjunto de registros que hacen posible la realización de cada una de las operaciones.

La mayoría de las acciones de la computadora son realizadas por la ALU. La ALU toma datos de los registros del procesador. Estos datos son procesados y los resultados de esta operación se almacenan en los registros de salida de la ALU. Otros mecanismos mueven datos entre estos registros y la memoria.

Una unidad de control controla a la ALU, al ajustar los circuitos que le señala a la ALU qué operaciones realizar.

ALU

La ALU es la parte del computador que realiza realmente las operaciones aritméticas y lógicas con los datos. El resto de los elementos del computador (UC, registros, memoria y E/S) están principalmente para suministrar datos a la ALU, a fin de que ésta los procese, y para recuperar los resultados.

Una ALU se basa en el uso de dispositivos lógicos digitales sencillos que pueden almacenar dígitos binarios y realizar operaciones lógicas booleanas elementales.

Los datos se presentan a la ALU en registros, y en registros se almacenan los resultados de las operaciones producidas por la ALU. Estos registros son posiciones de memoria temporal internas al procesador que están conectados a la ALU. La ALU puede también activar flags como resultado de una operación. Los valores de los indicadores se almacenan también en otro registro dentro del procesador. La UC proporciona las señales que gobiernan el funcionamiento de la ALU y la transferencia de datos dentro y fuera de la ALU.

4. Unidad Central de Procesamiento (CPU)

Organización de la CPU. Descripción de microprocesadores actuales. Modelo de ejecución de instrucciones. Ciclo de instrucción, fases. Comunicación CPU – memoria, dato y dirección. Interconexión de subsistemas, buses, ejemplos reales. Concepción de instrucción. Conjunto de instrucciones: operaciones, formato y modos de direccionamiento. Organización de registros. Lenguaje de máquina y assembly.

4.1 Organización de la CPU

Tanenbaum

La CPU es el “cerebro” de la computadora. Su función es ejecutar programas almacenados en la memoria principal buscando instrucciones y examinándolas para después ejecutarlas una tras otra. Sus componentes están conectados por un bus, que es una colección de alambres paralelos que transmiten direcciones, datos y señales de control. Los buses pueden ser externos a la CPU, cuando la conectan a la memoria y a los dispositivos de E/S, pero también internos, como se verá a continuación.

La CPU se compone de varias partes. La unidad de control (UC) se encarga de buscar instrucciones de la memoria principal y determinar su tipo. La unidad aritmética lógica (ALU) realiza operaciones como suma y AND booleano necesarias para ejecutar las instrucciones.

La CPU también contiene una memoria pequeña y de alta velocidad que sirve para almacenar resultados temporales y cierta información de control. Esta memoria se compone de varios registros, cada uno de los cuales tiene cierto tamaño y función.

El registro más importante es el contador de programa (PC), que apunta a la siguiente instrucción que debe buscarse para ejecutarse. Otro registro importante es el registro de instrucciones (IR), que contiene la instrucción que se está ejecutando.

Camino de datos: consiste en los registros (del 1 al 32), la ALU y varios buses que se conectan a los componentes. Los registros alimentan dos registros de entrada de la ALU. Estos registros contienen las entradas de la ALU mientras ésta está calculando. La ALU suma, resta y realiza otras operaciones simples con sus entradas, y produce un resultado en el registro de salida. El contenido de este registro de salida se envía a otro registro, que posteriormente se escribe en la memoria.

Casi todas las instrucciones pueden dividirse en una de dos categorías: **registro-memoria** o **registro-registro**. Las instrucciones registro-memoria permiten buscar palabras de la memoria a los registros, donde pueden utilizarse como entradas de la ALU en instrucciones subsecuentes, por ejemplo. Otras instrucciones registro-memoria permiten almacenar el contenido de un registro en la memoria.

La otra clase de instrucción es la de registro-registro. Una instrucción registro-registro típica busca dos operandos de los registros, los coloca en los registros de entrada de la ALU, realiza alguna operación con ellos y coloca el resultado en uno de los registros. El proceso de hacer pasar dos operandos por la ALU y almacenar el resultado se llama *ciclo del camino de datos* y es el corazón de casi todas las CPU. En

gran medida, este ciclo define lo que la máquina puede hacer. Cuanto más rápido es el ciclo del camino de datos, más rápidamente opera la máquina.

4.2 Características de las instrucciones máquina

El funcionamiento de la CPU está determinado por las instrucciones que ejecuta. Estas instrucciones se denominan instrucciones máquina. Al conjunto de instrucciones distintas que puede ejecutar la CPU se le denomina repertorio de instrucciones de la CPU.

Elementos de una instrucción máquina

Los elementos constitutivos de una instrucción son:

- **Código de operación:** especifica la operación a realizar. La operación se indica mediante un código binario, denominado “codop”.
- **Referencia a operandos fuente:** la operación puede implicar uno o más operandos fuente, es decir, operandos que son entradas para la instrucción.
- **Referencia al operando resultado:** la operación puede producir un resultado.
- **Referencia a la siguiente instrucción:** dice a la CPU de dónde captar la siguiente instrucción tras completarse la ejecución de la instrucción actual.

La siguiente instrucción a captar está en memoria principal o secundaria. Los operandos fuente y resultado pueden estar en alguna de las siguientes áreas:

- **Memoria principal o virtual:** debe indicarse su dirección.
- **Registro de la CPU:** si es un solo registro, la referencia es implícita, si es más de uno, debe indicarse el número de registro deseado en la instrucción.
- **Dispositivo de E/S:** la instrucción debe especificar el módulo y el dispositivo para la operación.

Representación de las instrucciones

Cada instrucción se representa mediante una secuencia de bits. La instrucción está dividida en campos, correspondientes a los elementos constitutivos de la misma. Durante su ejecución, la instrucción se escribe en un registro de instrucción (IR) de la CPU. La CPU debe ser capaz de extraer los datos de los distintos campos de la instrucción para realizar la operación requerida.

Los codops se representan mediante abreviaturas que indican la operación en cuestión, como ADD, SUB, DIV, etc. Los operandos también se representan simbólicamente, como ADD R, X.

4.3 Flujo de datos

Durante el ciclo de captación se lee una instrucción de la memoria. El PC contiene la dirección de la siguiente instrucción que hay que captar. Esta dirección se lleva a MAR y se pone en el bus de direcciones. La CU solicita una lectura de memoria, y el resultado se pone en el bus de datos, se copia en MBR y después se lleva a IR. Mientras tanto, PC se incrementa en uno como preparación para la siguiente captación.

Una vez concluido el ciclo de captación, la CU examina los contenidos de IR para determinar si contiene un campo de operando que use direccionamiento indirecto. Si es así, se lleva a cabo un ciclo indirecto. Se trata de un ciclo simple. Los N bits más a la derecha de MBR, que contienen la dirección de referencia, se transfieren a MAR. Entonces, la CU solicita una lectura de memoria para llevar la dirección del operando deseada a MBR.

Los ciclos de captación e indirecto son sencillos y predecibles. El ciclo de ejecución adopta muchas formas, ya que depende de cuál de las diversas instrucciones máquina esté en IR. Este ciclo puede implicar transferencias de datos entre registros, lectura o escritura de memoria o E/S, y/o la invocación de la ALU.

Del mismo modo que los ciclos de captación e indirecto, el ciclo de interrupción es simple y predecible. El contenido actual de PC tiene que ser salvado, de manera que la CPU pueda reanudar su actividad normal tras la interrupción. Así, el contenido de PC se transfiere a MBR para ser escrito en memoria. La posición de memoria especial reservada para este propósito se carga en MAR desde la CU. Podría ser, por ejemplo, un puntero de pila. PC se carga con la dirección de la rutina de interrupción. Como resultado, el siguiente ciclo de instrucción comenzará captando la dirección oportuna.

4.4 *Tipo de instrucciones*

El repertorio de instrucciones debe ser suficientemente amplio como para expresar cualquiera de las instrucciones de alto nivel. Teniendo esto presente, los tipos de instrucciones se clasifican de la siguiente manera:

- De procesamiento de datos: instrucciones aritméticas y lógicas.
- De almacenamiento de datos: instrucciones de memoria.
- De transferencia de datos: instrucciones de E/S.
- De control: instrucciones de comprobación y bifurcación.

Las instrucciones aritméticas proporcionan capacidad computacional para procesar datos numéricos. Las instrucciones lógicas operan sobre los bits de una palabra, el lugar de considerarlos como números, proporcionando capacidad para el procesamiento de cualquier otro tipo de datos que el usuario deba emplear. Estas operaciones se realizan principalmente con datos en registros de la CPU. Por lo tanto, debe haber instrucciones de memoria para transferir los datos entre la memoria y los registros. Las instrucciones de E/S se necesitan para transferir programas y datos a la memoria, y devolver los resultados de los cálculos al usuario. Las instrucciones de comprobación o test se emplean para comprobar el valor de una palabra de datos o el estado de un cálculo. Las de bifurcación se usan entonces para bifurcar a diferentes conjuntos de instrucciones dependiendo de la decisión tomada.

Número de direcciones

La mayoría de las CPU trabajan con instrucciones de una, dos o tres direcciones, siendo implícita la dirección de la instrucción siguiente (obtenida a partir del PC).

El número de direcciones por instrucción es una decisión de diseño. Menos direcciones por instrucción significa instrucciones más primarias, lo que requiere una CPU menos compleja. También da lugar a instrucciones más cortas. Por otra parte, los programas contienen más instrucciones, lo que normalmente supone mayor tiempo de ejecución y programas más largos y complejos. Hay también un umbral importante entre instrucciones de una y de múltiples direcciones. Con instrucciones de una sola dirección, el programador tiene generalmente a su disposición sólo un registro de uso general: el acumulador. Con instrucciones de múltiples direcciones suele disponerse de múltiples registros de uso general. Esto permite que algunas operaciones se realicen sólo con registros. Ya que los accesos a registros son más rápidos que a memoria, se acelera la ejecución. Por razones de flexibilidad y facilidad para utilizar varios registros, la mayoría de las máquinas emplean una combinación de instrucciones de dos y de tres instrucciones.

Diseño del repertorio de instrucciones

El diseño del repertorio afecta a muchos aspectos del computador. El repertorio define muchas de las funciones realizadas por la CPU y tiene, por tanto, un efecto significativo sobre la implementación de la misma. El repertorio es el medio que tiene el

programador para controlar la CPU, por lo tanto deben considerarse las necesidades de éste a la hora de diseñarlo.

Los aspectos más importantes de diseño son:

- **Repertorio de operaciones:** cuántas y qué operaciones considerar y de qué complejidad
- **Tipos de datos:** los distintos tipos de datos con los que se efectúan las operaciones.
- **Formatos de instrucciones:** longitud de la instrucción, número de direcciones, etc.
- **Registros:** número de registros de la CPU que pueden ser referenciados y uso.
- **Direccionamiento:** el modo mediante el que puede especificarse la dirección de un operando.

Tipos de operandos

Las instrucciones máquina operan con datos. Las categorías más importantes de datos son:

- **Direcciones:** ver direccionamiento. Son enteros sin signo.
- **Números:** enteros o en coma fija, en coma flotante y en decimal.
- **Caracteres:** ASCII, EBCDIC.
- **Datos lógicos:** representación orientada a bits.

Tipos de operaciones

- Transferencia de datos
- Aritméticas
- Lógicas
- Conversión
- Entrada – Salida
- Control del sistema
- Control de flujo (bifurcación, salto, llamada a procedimiento).

4.5 Lenguaje de máquina y ensamblador

Lenguaje de máquina

Los circuitos electrónicos de una computadora pueden reconocer y ejecutar directamente un conjunto limitado de instrucciones sencillas, y todos los programas tienen que convertirse en una serie de esas instrucciones para que la computadora puede ejecutarlos. Dicho grupo de instrucciones primitivas que una computadora puede ejecutar constituyen su *lenguaje máquina*, y permiten a las personas comunicarse con la computadora. Al diseñar una computadora, por lo general, se trata de hacer las instrucciones primitivas lo más básicas posibles (podrían ser, por ej., sumar dos números, verificar si un número es cero), en congruencia con el uso que se le piensa dar a la computadora y sus requisitos de desempeño, a fin de reducir su complejidad y el costo de los circuitos requeridos.

El lenguaje de máquina es el código del nivel ISA (*Instruction Set Architecture*, arquitectura del conjunto de instrucciones), que es el nivel 2 de una computadora multinivel (comúnmente llamado “nivel de máquina”). El nivel ISA es la interfaz entre los compiladores y el hardware. Para producir código en el nivel ISA, el escritor de compiladores tiene que conocer el modelo de memoria, los registros disponibles, los tipos de datos e instrucciones con que se cuenta, etc.

Lenguaje de Ensamble

Un lenguaje ensamblador puro es una representación simbólica de un programa en lenguaje de máquina subyacente, en el que cada enunciado produce exactamente una instrucción de máquina, es decir, existe una relación uno a uno entre las instrucciones de máquina y los enunciados del programa en lenguaje assembly.

El programador en lenguaje ensamblador solo tiene que recordar los nombres simbólicos (por ejemplo ADD, SUB, MUL, DIV) porque el programa ensamblador los traduce a instrucciones de máquina. Lo mismo se aplica a las direcciones. Los lenguajes de ensamblaje se distinguen de los lenguajes de alto nivel por:

- Su correspondencia uno a uno con enunciados en lenguaje máquina.
- Tener acceso a todas las características e instrucciones disponibles en la máquina objetivo. Todo lo que puede hacerse en lenguaje máquina puede hacerse en lenguaje ensamblador, pero muchas instrucciones, registros y características similares no están disponibles para el programador en lenguajes de alto nivel. Los lenguajes de programación de sistemas, como C, suelen ser un híbrido entre estos dos tipos.
- Un programa en lenguaje ensamblador solo puede correrse en una familia dada de computadoras, mientras que un programa escrito en un lenguaje de alto nivel posee el potencial para ejecutarse en muchas máquinas.
- Los programas en lenguaje ensamblador se traducen a lenguaje de máquina por programas denominados ensambladores. Los lenguajes de alto nivel también suelen traducirse (en algunos casos se interpretan) pero quienes lo hacen son compiladores.*

Ventajas del lenguaje ensamblador

- Usualmente se puede producir código mucho más pequeño y rápido que el que se puede producir en un lenguaje de alto nivel.
- Buen desempeño
- Acceso total a la máquina/hardware
- Útil para generar los programas de computadoras portátiles de escasos recursos, como las tarjetas inteligentes, procesadores incorporados a aparatos domésticos y asistentes digitales portátiles inalámbricos.

Casi todos los ensambladores son de dos pasadas. La primera pasada se dedica a construir una tabla de símbolos para etiquetas, literales e identificadores declarados explícitamente. La segunda pasada realiza la generación del código.

Casi todos los programas consisten en más de un procedimiento. Los ensambladores (y también los compiladores) traducen un procedimiento a la vez y lo guardan. Antes de que el programa pueda ejecutarse, es preciso encontrar todos los procedimientos traducidos y enlazarlos en forma correcta. Los programas que realizan estas funciones se denominan enlazador (linker). El ensamblador se encarga de ensamblar los procedimientos fuente, y el enlazador enlaza los módulos objeto.

Ensamblador

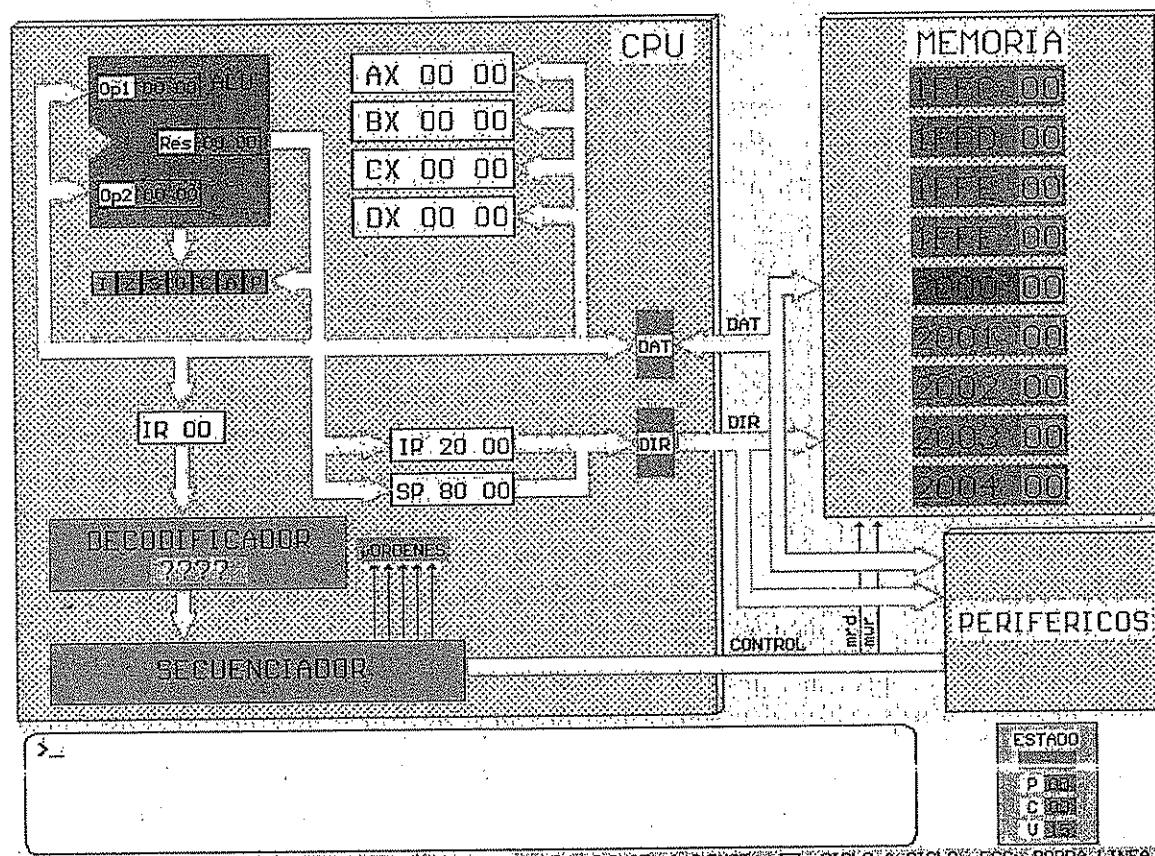
Las computadoras son máquinas diseñadas para ejecutar las instrucciones que se le indican, de manera de resolver problemas o hacer algún otro tipo de cosas. Dichas instrucciones básicas no suelen ser más complicadas que realizar una suma o resta, una comparación, leer un dato o escribir un resultado en la memoria. Además, lo normal es que cada instrucción esté almacenada mediante un código en la memoria, de forma que la computadora “ve” esos números y los entiende como instrucciones. También, estos códigos son específicos de cada tipo de CPU. A este conjunto de instrucciones codificadas se le llama lenguaje de máquina y es el único lenguaje que una computadora entiende.

Ahora bien, como programador, no resulta reconfortante pensar y escribir programas como una secuencia de números, de la manera en que la computadora los quiere ver, sino que es natural verlos como una secuencia de instrucciones. Para ello, se define un lenguaje, en principio, que es un mapeo directo de esos códigos a instrucciones comprensibles por un humano. A este lenguaje se lo denomina *assembly* o *lenguaje de ensamble* y al programa encargado de tomar programas escritos en assembly y generar los códigos que la computadora entenderá se lo denomina *assembler* o *ensamblador*. Como veremos, no todas las instrucciones de assembly se corresponden directamente con instrucciones de lenguaje de máquina, sino que varias son instrucciones para el ensamblador mismo.

El Simulador

Antes de comenzar con algunos ejemplos de programas escritos en assembly, veremos una breve descripción de un programa cuyo propósito es simular una CPU y mostrar los pormenores de su funcionamiento. Este programa se llama **MSX88** y simula

una computadora basada en una versión simplificada del célebre procesador 8086 de Intel, denominada SX88. Gráficamente se muestran los flujos de información existentes entre los diversos elementos que lo componen. Utiliza como plataforma de sistema operativo DOS, pudiéndose utilizar desde una ventana DOS bajo Windows. En la Figura siguiente se muestra la pantalla inicial.



En la Figura podemos distinguir los siguientes bloques:

- **CPU**
 - ALU
 - Registros **AX, BX, CX, DX, SP, IP, IR.**
 - Registro de **Flags**
 - **Decodificador**
 - **Secuenciador**
- **Memoria principal**
- **Periféricos**
- **Bus de datos y de direcciones**
- **Toma de entrada de comandos**

La CPU es la encargada de ejecutar un programa contenido en la memoria instrucción por instrucción.. La ALU es la encargada de ejecutar las operaciones aritméticas y lógicas entre los registros temporarios Op1 y Op2, dejando el resultado en Res. Dichas operaciones serán las dispuestas por la instrucción en ejecución Los

registros AX, BX, CX y DX son de uso general, 16 bits de longitud, se pueden dividir en 2 partes de 8 bits cada uno. Ejemplo: AX en AH y AL.

El registro IP (*instrucción pointer*) contiene la dirección de memoria de la próxima instrucción a ser ejecutada. El registro SP (*stack pointer*) contiene la dirección de memoria del tope de la pila. El registro de flags nos mostrará el estado de las banderas o flags luego de cada operación. Son 8 bits que indican el estado de las correspondientes 8 banderas. De estas 8 se utilizan:

- Bandera de cero: identificada por la letra Z.
- Bandera de overflow: identificada por la letra O.
- Bandera de carry/borrow: identificada por la letra C.
- Bandera de signo del número: identificada por la letra S.

La entrada de comandos es el lugar donde se introducirán los comandos del simulador una vez iniciada la aplicación. El bloque de periféricos engloba la entrada-salida del simulador.

4.6 Organización de los registros

Stallings

Dentro de la CPU hay un conjunto de registros que funciona como un nivel de memoria, por encima de la memoria principal y de la cache en la jerarquía. Los registros de la CPU son de dos tipos:

1. **Registros visibles para el usuario:** permiten minimizar las referencias a memoria principal cuando se optimiza el uso de registros.
2. **Registros de control y de estado:** son utilizados por la UC para controlar el funcionamiento de la CPU y por programas privilegiados del sistema operativo para controlar la ejecución de programas.

No existe separación bien definida entre estas categorías.

Registros visibles para el usuario

Un registro visible para el usuario es uno que puede ser referenciado por medio del lenguaje máquina que ejecuta la CPU. Se clasifican de la siguiente manera:

- **Uso general:** pueden ser asignados por el programador a diversas funciones. Cualquier registro de uso general puede contener el operando para cualquier código de operación. Sin embargo, existen restricciones, por ejemplo puede haber registros específicos para operaciones en coma flotante y operaciones de pila. En algunos casos pueden ser utilizados para funciones de direccionamiento, en otros hay una separación parcial o total entre registros de datos y de direcciones.
- **Datos:** se usan únicamente para contener datos y no se pueden emplear en el cálculo de una dirección de operando.
- **Direcciones:** pueden ser de uso más o menos general, o pueden estar dedicados a un modo de direccionamiento particular. Dentro de éstos se encuentran los

punteros de segmentos (contienen la dirección base de segmento), los registros índice (se usan para direccionamiento indexado o autoindexado) y los punteros de pila (direcciónamiento a pila visible al usuario –implícito–).

- **Códigos de condición** (flags): son bits fijados por el hardware de la CPU como resultado de una operación. Forman parte de un registro de control. Son parcialmente visibles al usuario y no pueden ser alterados.

Registros de control y de estado

La mayoría no son visibles para el usuario, algunos lo son a instrucciones máquina ejecutadas en un modo de control o de Sistema Operativo.

Son esenciales cuatro registros para la ejecución de una instrucción:

- **Contador de Programa** (PC): contiene la dirección de la instrucción a captar.
- **Registro de Instrucción** (IR): contiene la instrucción captada más recientemente.
- **Registro de dirección de memoria** (MAR): contiene la dirección de una posición de memoria.
- **Registro intermedio de memoria** (MBR): contiene la palabra de datos a escribir en memoria, o la palabra leída más recientemente.

Los cuatro registros se usan para la transferencia de datos entre la CPU y la memoria. Dentro de la CPU, los datos tienen que ofrecerse a la ALU para su procesamiento. La ALU puede tener acceso directo a MBR y a los registros visibles para el usuario. Como alternativa, puede haber registros intermedios adicionales entorno a la ALU que sirven como registros de E/S de la ALU, e intercambian datos con MBR y los registros visibles para el usuario.

Todos los diseños de CPU incluyen un registro o un conjunto de registros, conocidos como *palabra de estado de programa* (PSW), que contiene información de estado, normalmente códigos de condición. Entre los indicadores comunes se incluyen:

- Signo
- Cero
- Acarreo
- Igual
- Desbordamiento
- Interrupciones habilitadas/inhabilitadas
- Supervisor (modo supervisor o modo usuario)

4.7 Direccionamiento

Como los campos de direcciones en un formato de instrucciones usual están bastante limitados, es deseable poder referenciar un rango elevado de posiciones de memoria principal o virtual. Para esto se emplean las técnicas de direccionamiento. Para explicarlas, usamos la siguiente notación:

- A = contenido de un campo de dirección en la instrucción.
- R = contenido de un campo de dirección en la instrucción que referencia a un registro.
- EA = dirección real (efectiva) de la posición que contiene el operando que se referencia.
- (X) = Contenido de la posición X.

Direccionamiento inmediato

El operando está presente en la propia instrucción:

OPERANDO=A

Este modo puede usarse para definir y utilizar constantes, o para fijar valores iniciales de variables.

La ventaja de este modo es que, una vez captada la instrucción, no se requiere una referencia a memoria para obtener el operando, ahorrándose así un ciclo de memoria o de caché. La desventaja es que el tamaño del número está restringido a la longitud del campo de direcciones que, en su mayoría, es pequeño comparado con la longitud de la palabra.

Ejemplo: MOV AX, 6H.

Direccionamiento directo

El campo de direcciones contiene la dirección efectiva del operando:

EA=A

Sólo requiere una referencia a memoria y no necesita ningún cálculo especial. La limitación obvia es que proporciona un espacio de direcciones restringido.

Ejemplo:

```
ORG 1000h
NUM1 Db 10h
NUM2 Db 20h
ORG 2000h
MOV cl, NUM1
Mov al, NUM2
ADD AL, CL
HLT
END.
```

Direccionamiento indirecto

El campo de direcciones referencia a la dirección de una palabra de memoria que contiene la dirección completa del operando:

EA=(A)

La ventaja es que, para una longitud de palabra de N bits, se dispone de un espacio de direcciones de 2^N . La desventaja es que la ejecución de la instrucción requiere dos referencias a memoria para captar el operando: una para captar su dirección y otra para obtener su valor.

Ejemplo:

```
DIR1 db 1001h
MOV AX, [DIR1]
HLT
END.
```

Direccionamiento de registros

Es similar al directo, aunque ahora el campo de direcciones referencia a un registro:

EA = R

Pueden referenciarse un total de 8 ó 16 registros de uso general.

Las ventajas de este modo son que sólo es necesario un campo pequeño de direcciones en la instrucción y no se requieren referencias a memoria. A su vez, el tiempo de acceso a un registro interno de la CPU es mucho menos que a la memoria principal. La desventaja es que el espacio de direcciones está muy limitado.

Direccionamiento indirecto con registro

Es análogo al indirecto, aunque el campo de direcciones hace referencia a un registro:

EA = (R)

Las ventajas y limitaciones de este modo son básicamente las mismas que se tienen en el indirecto. La limitación del espacio se supera haciendo que el campo de direcciones refiera a una posición de una palabra completa (un registro completo) que contenga la dirección. Además, este modo emplea una referencia menos a memoria que el direccionamiento indirecto.

Ejemplo:

```
MOV AX, [BX]
```

Direccionamiento con desplazamiento

Combina las posibilidades de los direccionamientos directo e indirecto con registro:

EA = A + (R)

Requiere que las instrucciones tengan dos campos de direcciones, al menos uno de ellos explícito. El valor contenido en uno de los campos de direcciones (valor = A) se utiliza directamente, el otro campo de direcciones, o una referencia implícita definida por el codop, se refiere a un registro cuyo contenido se suma a A para generar la dirección efectiva.

Tres de los usos más comunes del direccionamiento con desplazamiento son:

1. **Direccionamiento relativo:** el registro direccionado implícitamente es el PC. Es decir, la dirección de instrucción actual se suma al campo de direcciones para producir el valor EA. La dirección efectiva es un desplazamiento relativo a la dirección de la instrucción. Este tipo de desplazamiento aprovecha el concepto de localidad.
2. **Direccionamiento con registro base:** el registro referenciado contiene una dirección de memoria, y el campo de dirección contiene un desplazamiento desde dicha dirección. La referencia a registro puede ser implícita o explícita. Este direccionamiento también aprovecha la localidad de las referencias a memoria.
3. **Indexado:** el campo de dirección referencia una dirección de memoria principal, y el registro referenciado contiene un desplazamiento positivo desde esa dirección. Un uso importante del indexado es como mecanismo eficiente para ejecutar operaciones iterativas.

Dado que los registros índice se usan normalmente para tales tareas iterativas, es normal incrementarlos o decrementarlos tras cada referencia. Ya que ésta es una operación común, algunos sistemas la efectúan automáticamente como parte del ciclo de instrucción. Esto se denomina auto-indexado, que puede describirse como sigue:

$$\begin{aligned} EA &= A + (R) \\ (R) &\leftarrow (R) + 1 \end{aligned}$$

Si la indexación se realiza después de la indirección se denomina postindexado:

$$EA = (A) + (R)$$

Primero, el contenido del campo de direcciones se emplea para acceder a la posición de memoria que contiene una dirección directa. Esta dirección se indexa entonces mediante el valor del registro. Esta técnica es útil para acceder a uno de entre un número de bloques de datos con un formato fijo.

Con pre-indexado, la indexación se realiza antes de la indirección:

$$EA = (A + (R))$$

La dirección se calcula como en el caso del indexado simple. No obstante, en este caso la dirección calculada no contiene al operando, sino la dirección del operando.

Direccionamiento de pila

El valor del puntero tiene la dirección del tope de pila. Alternativamente, los dos elementos del tope pueden residir en registros de la CPU, en cuyo caso el puntero de pila hace referencia al tercer elemento de la pila. El puntero de pila se mantiene en un registro. Así, las referencias a posiciones de la pila en memoria son, de hecho, direcciones de acceso indirecto con registro. El modo de direccionamiento de pila es una forma de direccionamiento implícito. Las instrucciones máquina no necesitan incluir una referencia a memoria, sino que operan implícitamente con la cabecera de la pila.

Ejemplo

```
POP AX  
PUSH AX  
POPF AX  
PUSHF AX
```

4.8 Formato de instrucciones

Un formato de instrucciones define la descripción en bits en una instrucción en términos de las distintas partes que la componen. Un formato de instrucción debe incluir un codop e, implícita o explícitamente, ninguno o algunos operandos. Cada operando explícito se referencia usando uno de los modos de direccionamiento. El formato debe, implícita o explícitamente, indicar el modo para cada operando. En la mayoría de los repertorios de instrucciones se emplea más de un formato de instrucción. A continuación se detallan algunos aspectos de diseño.

Longitud de instrucción

Esta decisión afecta y se ve afectada por el tamaño de la memoria, su organización, la estructura de buses, la complejidad de la CPU y la velocidad de la CPU. Esta decisión define la riqueza y flexibilidad de la máquina desde el punto de vista del programador de lenguaje ensamblador.

Asignación de los bits

Existe un compromiso entre el número de codops y la capacidad de direccionamiento. Los siguientes factores, relacionados entre sí, afectan a la definición del uso dado a los bits de direccionamiento:

- **Número de modos de direccionamiento:** implícito o determinar el número.
- **Número de operandos:** dos operandos
- **Registros frente a memoria:** 8 a 32.
- **Número de conjunto de registros:** un solo banco de registros o más especializados.
- **Rango de direcciones:** se relaciona con el número de bits de direccionamiento.
- **Granularidad de las direcciones.**

Instrucciones de longitud variable

Usar estas instrucciones hace fácil proporcionar un amplio repertorio de codops de longitud variable. El direccionamiento puede ser más flexible, con varias combinaciones de referencias a registros y a memoria, así como de modos de direccionamiento. Con instrucciones de longitud variable, estas múltiples variaciones pueden proporcionarse de manera eficiente y compacta.

El precio a pagar por las instrucciones de longitud variable es el aumento de complejidad de la CPU. La disminución del precio del hardware, el uso de microprogramación y un aumento general en el conocimiento de los principios de diseño de la CPU contribuyen todos a hacer que el precio a pagar sea pequeño.

El uso de instrucciones de longitud variable no elimina el deseo de que todas las longitudes de instrucciones estén directamente relacionadas con la longitud de la palabra. Ya que la CPU no conoce la longitud de la instrucción que va a captar, una estrategia usual es captar un número de bytes o de palabras igual, al menos, al tamaño de la instrucción más larga. Esto significa que a veces se captan varias instrucciones.

4.9 El ciclo de instrucción

Se denomina así al procesamiento que requiere una instrucción. La ejecución de un programa se para, sólo si la máquina se desconecta, se produce algún tipo de error, o ejecuta una instrucción del programa que detiene al computador.

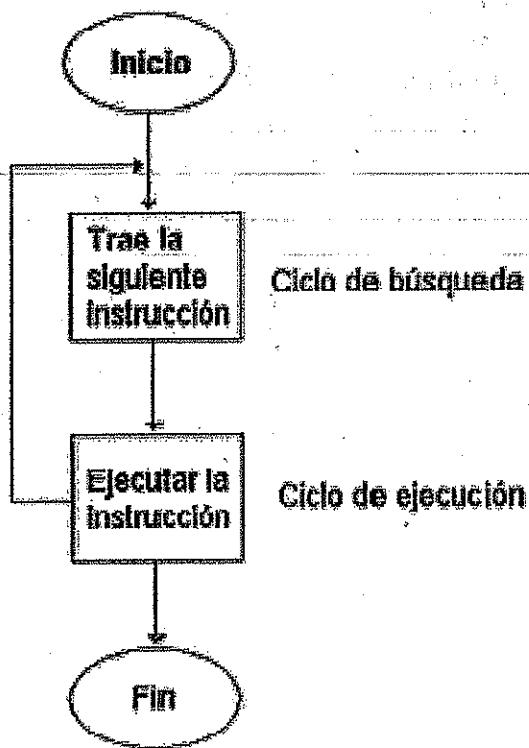
Los ciclos de captación y ejecución

Al comienzo de cada ciclo de instrucción, la CPU capta una instrucción de memoria. Se usa el PC para seguir la pista de la instrucción que debe captarse a continuación. A no ser que se indique otra cosa, la CPU siempre incrementa el PC después de captar cada instrucción, de forma que captará la siguiente de la secuencia.

La instrucción captada se almacena en IR. La instrucción se escribe utilizando un código binario que especifica la acción que debe realizar la CPU. La CPU interpreta la instrucción y lleva a cabo la acción requerida. En general, puede ser de cuatro tipos:

1. **Procesador-memoria:** deben transferirse datos desde la CPU a la memoria, o desde la memoria a la CPU.
2. **Procesador-E/S:** deben transferirse datos a o desde el exterior mediante transferencias entre la CPU y un módulo de E/S.
3. **Procesamiento de datos:** la CPU ha de realizar alguna operación aritmética o lógica con los datos.
4. **Control:** una instrucción puede especificar que la secuencia de ejecución se altere (por ejemplo, un salto).

La ejecución de una instrucción puede implicar una combinación de estas acciones.

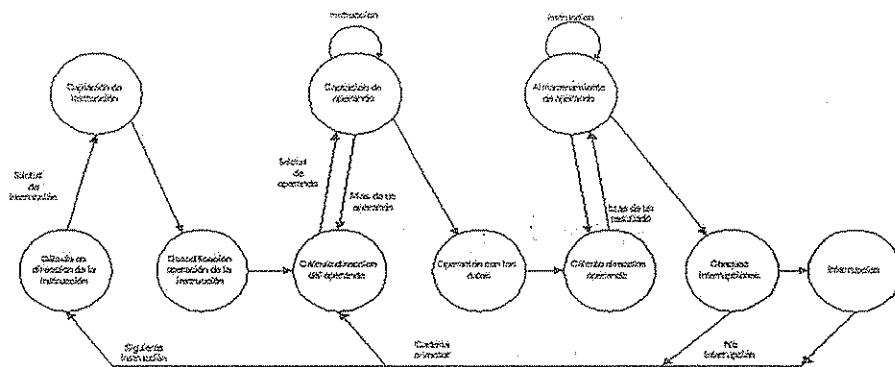


Como el ciclo de ejecución de una instrucción particular puede ocasionar más de una referencia a memoria y, además, en lugar de referencias a memoria, una instrucción puede especificar una operación de E/S, se extiende el diagrama de estados del ciclo de instrucción de una forma más detallada. Para un ciclo de instrucción dado, algunos estados pueden no darse y otros pueden visitarse más de una vez. Los estados se describen a continuación:

- **Cálculo de la dirección de la instrucción (iac):** determina la dirección de la siguiente instrucción a ejecutar.
- **Captación de la instrucción (if):** la CPU lee la instrucción desde su posición en memoria.
- **Decodificación de la operación indicada en la instrucción (iod):** analiza la instrucción para determinar el tipo de operación a realizar y el/los operandos a utilizar.
- **Cálculo de la dirección del operando (oac):** si la instrucción implica una referencia a un operando en memoria o disponible mediante E/S, determina la dirección del operando.
- **Captación del operando (of):** capta el operando desde memoria o se lee desde el dispositivo de E/S.
- **Operación con los datos (do):** realiza la operación indicada en la instrucción.
- **Almacenamiento del operando (os):** escribe el resultado en memoria o lo saca a través de un dispositivo de E/S.

Los estados de la parte superior de la figura siguiente ocasionan intercambios entre la CPU y la memoria o un módulo de E/S. Los estados en la parte inferior sólo ocasionan operaciones internas a la CPU. El estado oac aparece dos veces, puesto que una instrucción puede ocasionar una lectura, una escritura o ambas cosas. No obstante,

la acción realizada en ese estado es la misma en ambos casos y, por eso, sólo se necesita un único identificador de estado.



4.10 Interrupciones

Las interrupciones proporcionan una forma de mejorar la eficiencia del procesador. Existen diferentes clases de interrupciones:

- **Programa:** generadas por alguna condición que se produce como resultado de la ejecución de una instrucción, como la división por cero.
- **Temporización:** generadas por un temporizador interno al procesador. Permite al sistema operativo realizar ciertas funciones de manera regular.
- **E/S:** generadas por un controlador de E/S para indicar la finalización sin problemas de una operación o para avisar de ciertas condiciones de error.
- **Fallo de hardware:** generadas por un fallo tal como un error de paridad en la memoria.

Las interrupciones y el ciclo de instrucción

Con el uso de interrupciones el procesador puede dedicarse a ejecutar otras instrucciones mientras una operación de E/S está en curso. Cuando el procesamiento de la interrupción se completa, la ejecución prosigue. Así, el programa de usuario no tiene que incluir ningún código especial para posibilitar las interrupciones; el procesador y el sistema operativo son los responsables de detener el programa y después permitir que prosiga desde el mismo punto.

Para permitir el uso de interrupciones, se añade un ciclo de interrupción al ciclo de instrucción. En el ciclo de interrupción, el procesador comprueba si se ha generado una interrupción, indicada por la presencia de una señal de interrupción. Si no hay señales pendientes, el procesador continúa con el ciclo de captación y accede a la siguiente instrucción del programa en curso. Si hay una interrupción pendiente, el procesador hace lo siguiente:

1. **Suspende la ejecución** del programa en curso y **guarda su contexto**. Esto significa almacenar la dirección de la siguiente instrucción y cualquier otro dato relacionado con la actividad en curso del procesador.
2. **Carga el contador de programa** con la dirección de comienzo de una rutina de gestión de interrupción.

A continuación, el procesador prosigue con el ciclo de captación y accede a la primera instrucción del programa de gestión de interrupción, que dará servicio a la interrupción.

Interrupciones múltiples

Para tratar este tipo de interrupciones se pueden seguir dos alternativas. La primera es desactivar las interrupciones mientras se está procesando una interrupción. Una interrupción inhabilitada significa que el procesador puede y debe ignorar la señal de petición de interrupción. Si se produce una interrupción en ese momento, generalmente se mantiene pendiente, y será examinada por el procesador una vez que éste haya activado las interrupciones. Este enfoque tiene un inconveniente, que es que no tiene en cuenta la prioridad relativa, ni las solicitudes con un tiempo crítico.

Una segunda alternativa consiste en definir prioridades para las interrupciones y permitir que una interrupción de prioridad más alta pueda interrumpir a una de prioridad menor.

El ciclo indirecto

La ejecución de una instrucción puede involucrar a uno o más operandos en memoria, cada uno de los cuales requiere un acceso a memoria. Además, si se usa direccionamiento indirecto serán necesarios accesos a memoria adicionales.

Podemos considerar la captación de direcciones indirectas como un subciclo de instrucción. La principal línea de actividad consiste en alternar las actividades del ciclo de captación y ejecución de instrucciones. Después de que una instrucción sea captada, ésta es examinada para determinar si implica algún direccionamiento indirecto. Tras la ejecución se puede procesar una interrupción antes de la captación de la siguiente instrucción.

Una vez que la instrucción es captada, deben identificarse sus campos de operando. Se capta entonces de la memoria cada operando de entrada, pudiendo requerir este proceso direccionamiento indirecto. Los operandos ubicados en registros no necesitan ser captados. Una vez que se ejecuta la operación, puede ser necesario un proceso similar para almacenar el resultado en la memoria principal.

4.11 Estructuras de interconexión

Son el conjunto de líneas que conectan los módulos de un computador (procesador, memoria y E/S). El diseño de dicha estructura dependerá de los intercambios que deban producirse entre los módulos. Los tipos de intercambio son:

- **Memoria:** un módulo de memoria está constituido por N palabras de la misma longitud. A cada palabra se le asigna una única dirección numérica (de 0 a N-1). Una palabra de datos puede leerse o escribirse en la memoria. La posición de memoria para la operación se especifica mediante una dirección.

- **Módulo de E/S:** la E/S es funcionalmente similar a la memoria. Hay dos tipos de operaciones: leer y escribir. Además, un módulo puede controlar más de un dispositivo externo (puerto), al cual se le asigna una dirección a cada uno. Existen líneas de datos externas para la entrada y la salida de datos por un dispositivo externo. Por último, un módulo de E/S puede enviar señales de interrupción al procesador.
- **Procesador:** lee instrucciones y datos, escribe datos una vez que los ha procesado, y usa ciertas señales para controlar el funcionamiento del sistema. También puede recibir señales de interrupción.

La estructura de interconexión debe dar cobertura a los siguientes tipos de transferencia:

1. Memoria a procesador
2. Procesador a memoria
3. E/S a procesador
4. Memoria a E/S y viceversa

Interconexión con buses

Un bus es un camino de comunicación entre dos o más dispositivos. Al bus se conectan varios dispositivos y cualquier señal transmitida por uno de esos dispositivos está disponible para que los otros dispositivos conectados al bus puedan acceder a ella. Sólo un dispositivo puede transmitir con éxito en un momento dado.

Estructura del bus

El bus del sistema (que conecta procesador, memoria y E/S) está constituido por entre 50 y 100 líneas, que se pueden clasificar en:

- **Líneas de datos:** transmiten datos entre los módulos del sistema. El conjunto se denomina bus de datos. Consta de 8, 16 ó 32 líneas (anchura), que determinan cuántos bits se pueden transferir al mismo tiempo.
- **Líneas de dirección:** se usan para designar la fuente o el destino del dato situado en el bus de datos. La anchura del bus de direcciones determina la máxima capacidad de memoria posible.
- **Líneas de control:** se usan para controlar el acceso y el uso de las líneas de datos y de direcciones, ya que son compartidas. Algunas líneas típicas son:
 - Escritura en memoria
 - Lectura de memoria
 - Escritura de E/S
 - Lectura de E/S
 - Transferencia reconocida
 - Petición de bus
 - Cesión de bus
 - Petición de interrupción
 - Interrupción reconocida
 - Reloj
 - Inicio

Funcionamiento del bus

Si un módulo desea enviar un dato a otro debe hacer dos cosas: 1. Obtener el uso del bus y 2. Transferir el dato a través del bus. Si un módulo desea pedir un dato a otro módulo, debe 1. Obtener el uso del bus y 2. Transferir la petición al otro módulo mediante las líneas de control y dirección apropiadas. Después debe esperar a que el segundo módulo envíe el dato.

Jerarquía de buses

Si se conecta un gran número de dispositivos al bus, las prestaciones pueden disminuir por dos causas principales:

- A más dispositivos conectados al bus, mayor es el retardo de propagación, el cual determina el tiempo que necesitan los dispositivos para coordinarse en el uso del bus.
- El bus puede convertirse en un cuello de botella a medida que las peticiones de transferencia acumuladas se aproximan a la capacidad del bus. Esto se resuelve aumentando el ancho del bus, aunque un único bus ya no se usa.

Por consiguiente, se usan varios buses organizados jerárquicamente. En una estructura típica, hay un bus local que conecta el procesador a una memoria caché, y al que pueden conectarse también uno o más dispositivos locales. El controlador de caché conecta la caché no sólo al bus local, sino también al de sistema, donde se conectan todos los módulos de memoria principal.

Los controladores de E/S se conectan a un bus de expansión. La interfaz del bus de expansión regula las transferencias de datos entre el bus del sistema y los controladores conectados al bus de expansión, lo que aísla el tráfico de información con el de la memoria y el procesador.

Elementos de diseño de un bus

Los parámetros que clasifican los buses son:

- **Tipo de bus:** se divide en dos tipos genéricos:
 - **Dedicadas:** está permanentemente asignada a una función o a un subconjunto físico de componentes del computador.
 - **Multiplexadas:** mismas líneas para diferentes usos.
- **Método de arbitraje:** se clasifican en:
 - **Centralizado:** un árbitro es el responsable de asignar tiempos en el bus.
 - **Distribuido:** cada módulo dispone de lógica para controlar el acceso y los módulos actúan conjuntamente para compartir el bus.
- **Temporización:** hace referencia a la forma en la que se coordinan los eventos en el bus. Se clasifica en:
 - **Síncrona:** la presencia de un evento en el bus está determinada por un reloj.
 - **Asíncrona:** la presencia de un evento en el bus es consecuencia y depende de que se produzca un evento previo.

- **Anchura del bus:** cuanto más ancho es el bus de datos, mayor es el número de bits que se transmiten a la vez. Cuanto más ancho es el bus de direcciones, mayor es el rango de posiciones a las que se puede hacer referencia.
- **Tipos de transferencia de datos:** todos los buses permiten tanto transferencias de lecturas como de escrituras. En ciertos buses también son posibles algunas operaciones combinadas, como lectura-modificación-escritura o lectura-después-de-escritura.

PCI

El bus PCI es un bus de ancho de banda elevado, independiente del procesador, que se puede usar como bus de periféricos o bus para una arquitectura de entreplanta. Está diseñado para permitir una cierta variedad de configuraciones basadas en microprocesadores, incluyendo sistemas tanto de uno como de varios procesadores. Por consiguiente, proporciona un conjunto de funciones de uso general. Utiliza temporización síncrona y un esquema de arbitraje centralizado.

4.12 Descripción de microprocesadores actuales

El procesador Pentium

Organización de los registros

Incluye los siguientes registros:

- **Generales:** ocho registros de 32 bits, algunos de uso especial.
- **De segmento:** seis de 16bits. Indexan tablas de segmento. El registro de segmento de código (CS) referencia el segmento que contiene la instrucción que se está ejecutando. El registro de segmento de pila (SS) referencia el segmento que contiene una pila visible para el usuario. Los demás registros (DF, ES, FS y GS) permiten al usuario referenciar, al mismo tiempo, hasta cuatro segmentos de datos distintos.
- **Indicadores:** el registro EFLAGS contiene los códigos de condición y diversos bits de modo.
- **Puntero de instrucción:** contiene la dirección de la instrucción en curso.

Hay también registros dedicados específicamente a la unidad de coma flotante:

- **Numéricos:** cada registro contiene un número en coma flotante de 80 bits de precisión ampliada y 8 que funcionan como una pila.
- **De control:** tiene 16 bits que controlan el funcionamiento de la unidad de coma flotante, precisión simple, doble o ampliada, y bits para habilitar e inhabilitar diversas condiciones de excepción.
- **De estado:** de 16 bits contiene bits que reflejan el estado presente de la unidad de coma flotante, códigos de condición que informan sobre el resultado de la última operación, e indicadores de excepción.
- **Palabra de etiquetas:** de 16 bits contiene una etiqueta de 2 bits para cada registro numérico de coma flotante, que indica la naturaleza de los contenidos

del registro correspondiente. Los cuatro valores posibles son válido, cero, especial (NaN, infinito, desnormalizado) y vacío.

El procesador PowerPC

Organización de los registros

La unidad de coma fija incluye los siguientes:

- **Generales:** hay 32 registros de 64bits de uso general. Se pueden utilizar para cargar, almacenar y manipular operandos de datos, y también se pueden usar para direccionamiento indirecto a través de registro.
- **Registro de excepción (XER):** incluye 3 bits que informan sobre excepciones en operaciones aritméticas con enteros.

La unidad de coma flotante contiene otros registros visibles para el usuario:

- **Generales:** hay 32 registros de uso general de 64 bits, usados para todas las operaciones de coma flotante.
- **Registro de estado y control de coma flotante (FPSCR):** de 32 bits contiene bits que controlan el funcionamiento de la unidad de coma flotante, y bits que guardan el estado resultante de operaciones de coma flotante.

La unidad de procesamiento de saltos contiene estos registros visibles para el usuario:

- **Registro de condición:** consta de 8 campos de código de condición de 4bits.
- **Registro de enlace:** puede usarse en una instrucción de bifurcación condicional para el direccionamiento indirecto de la dirección destino.
- **Cuenta:** puede usarse para controlar iteraciones de bucles. Se decrementa cada vez que es examinado por una instrucción de bifurcación condicional.

5. Memoria

Tipos de memorias, clasificación. Parámetros característicos, tamaño, tiempo de acceso, costo, otros. Memoria principal, formas de organización. Memoria secundaria, organización y formato de datos. Organización jerárquica de la memoria. Dispositivos de almacenamiento externo, disco, cinta, disco óptico, otros. Múltiples unidades de discos (RAID).

5.1 Conceptos básicos sobre sistemas de memoria de computadores

Características de los sistemas de memoria

Las características más importantes de los sistemas de memoria son:

Ubicación

Puede ser interna o externa. La memoria interna suele identificarse con la memoria principal. Sin embargo, el procesador necesita su propia memoria local en forma de registros y la CU del procesador también usa su propia memoria interna. La memoria externa consta de dispositivos periféricos de almacenamiento, tales como discos y cintas.

Capacidad

Para memorias internas se expresa en bytes o palabras (8, 16 o 32 bits). Para memorias externas se expresa en bytes.

Unidad de transferencia

Es el número de bits que se leen o escriben en memoria a la vez. Para memorias internas es igual al número de líneas de E/S de datos del módulo de memoria (una palabra o una unidad direccionable, generalmente). Para memorias externas, los datos se transfieren en bloques.

Método de acceso

Incluye las siguientes variables:

- Acceso secuencial: cintas
- Acceso directo: disco
- Acceso aleatorio: memoria principal y algunas caché
- Asociativa: memorias caché

Prestaciones

Se utilizan tres parámetros de medida de prestaciones:

- Tiempo de acceso: para memorias de acceso aleatorio es el tiempo que tarda en realizarse una operación de lectura o escritura. Para otras memorias, es el

tiempo de acceso en que se tarda en situar el mecanismo de lectura/escritura en la posición deseada.

- **Tiempo de ciclo de memoria:** se aplica a las memorias de acceso aleatorio y consiste en el tiempo de acceso y algún tiempo más que se requiere, antes de que pueda iniciarse un segundo acceso a memoria.
- **Velocidad de transferencia:** es la velocidad a la que pueden transferirse los datos a, o desde, una unidad de memoria.

Dispositivos físicos

Las más comunes son las memorias sémiconductoras, las memorias de superficie magnética, utilizadas para discos y cintas, y las memorias ópticas y magneto-ópticas.

Características físicas

Volátiles o no volátiles.

Organización

Es la disposición o estructura física en bits para formar palabras.

5.2 Jerarquía de memoria

A medida que descendemos en la jerarquía de memoria, disminuye el coste por bit, aumenta la capacidad y crece el tiempo de acceso. Sería deseable poder utilizar sólo la memoria más rápida, pero al ser la más costosa, se llega a un compromiso entre el tiempo de acceso y coste, empleando más cantidad de memoria más lenta. La estrategia a seguir consiste en organizar los datos y los programas en memoria de manera que las palabras de memoria necesarias estén normalmente en la memoria más rápida.

En todo el espectro de posibles tecnologías se cumplen las siguientes relaciones:

1. A menor tiempo de acceso, mayor coste por bit.
2. A mayor capacidad, menor coste por bits.
3. A mayor capacidad, mayor tiempo de acceso.



Cuando se desciende en la jerarquía ocurre:

1. Disminuye el coste por bit
2. Aumenta la capacidad
3. Aumenta el tiempo de acceso
4. Disminuye la frecuencia de accesos a la memoria por parte del procesador.

La clave del éxito de esta organización está en el último ítem: la disminución de la frecuencia de acceso.

Empleando diversas tecnologías se tiene todo un espectro de sistemas de memoria que satisfacen las condiciones 1 y 3. Afortunadamente, la cuarta condición es también generalmente válida. La base para la validez de la condición 4 es el principio conocido como localidad de las referencias

Se denomina cercanía de referencias al agrupamiento de las lecturas de memoria por medio de la CPU. Las mismas, ya sean para instrucciones o para leer datos, se mantienen por lo general dentro de grupos de direcciones relativamente cercanas entre sí. Esto se da porque los programas normalmente cuentan con un cierto número de bucles y subrutinas iterativas. Una vez dentro de una de estas estructuras, se producirán referencias repetidas a un pequeño conjunto de instrucciones. Las agrupaciones de uso con el tiempo son variables, pero considerando un período corto de tiempo se mantienen fijas.

Cercanía de referencias también implica que para que un programa ejecute no son necesarias todas sus páginas cargadas en la memoria principal. Si en un período corto de tiempo el procesador referencia direcciones cercanas de memoria, un proceso puede ejecutar varias instrucciones con sólo algunas partes de su código en memoria principal. Esto permite tener más procesos en memoria listos para correr, como así también evita tener que cargar y descargar sus páginas en el momento de un intercambio, perdiendo ciclos del procesador.

5.3 Localidad de referencias

La localidad de las referencias, también conocida como el principio de localidad, es un fenómeno según el cual, basándonos en el pasado reciente de un programa podemos predecir con una precisión razonable qué instrucciones y datos utilizará en un futuro próximo.

Los casos más importantes de localidad son la localidad espacial, la localidad secuencial y la localidad temporal.

Localidad Temporal

Si en un momento una posición de memoria particular es referenciada, entonces es muy probable que la misma ubicación vuelva a ser referenciada en un futuro cercano.

Existe proximidad temporal entre las referencias adyacentes a la misma posición de memoria. En este caso es común almacenar una copia de los datos referenciados en caché para lograr un acceso más rápido a ellos.

Localidad Espacial

Si una localización de memoria es referenciada en un momento concreto, es probable que las localizaciones cercanas a ella sean también referenciadas pronto. Existe localidad espacial entre las posiciones de memoria que son referenciadas en momentos cercanos. En este caso es común estimar las posiciones cercanas para que estas tengan un acceso más rápido.

Localidad Secuencial

Las direcciones de memoria que se están utilizando suelen ser contiguas. Esto ocurre porque las instrucciones se ejecutan secuencialmente. Para obtener beneficios de la gran frecuencia con la que ocurren casos de localidad espacial o temporal, muchos sistemas de memoria utilizan una jerarquía de niveles de memoria.

5.3 Memoria principal semiconductor

Tipos de memorias semiconductoras de acceso aleatorio

RAM

Es la más común. Una característica distintiva es que es posible, tanto leer datos, como escribir rápidamente nuevos datos en ella. A su vez, es volátil. Las tecnologías de RAM se dividen en:

- **Dinámica:** está hecha con celdas que almacenan los datos como cargas en condensadores. Ya que los condensadores tienen una tendencia natural a descargarse, las RAM dinámicas requieren refrescos periódicos para mantener memorizados los datos.
- **Estática:** los valores binarios se almacenan utilizando configuraciones de puertas que forman biestables. Una RAM estática retendrá sus datos en tanto se mantenga alimentada.

Las RAM estáticas son más rápidas y menos densas que las dinámicas.

- **ROM:** contiene un patrón permanente de datos que no puede alterarse. Sus aplicaciones son: la microporgramación, subrutinas de biblioteca para funciones de uso frecuente, programas del sistema y tablas de funciones.
- **PROM:** son no volátiles y pueden grabarse una sola vez. Proporcionan flexibilidad y comodidad.
- **Memorias de sobre-todo-lectura:** hay tres formas:
 - **EPROM:** pueden modificarse múltiples veces y retienen su contenido indefinidamente. Una EPROM es más costosa que una PROM pero se puede actualizar muchas veces.
 - **EEPROM:** se puede escribir en cualquier momento sin borrar su contenido anterior. Son más costosas que las EPROM y menos densas (menos bits por chip).
 - **Memoria flash:** puede borrarse entera en unos cuantos segundos. Además, es posible borrar sólo bloques concretos de memoria, aunque no a nivel de byte.

Organización

El elemento básico de una memoria semiconductor es la celda de memoria. Todas las celdas comparten ciertas propiedades:

- Presentan dos estados estables que pueden emplearse para representar 1 y 0.
- Puede escribirse en ellas (al menos una vez) para fijar su contenido.
- Pueden leerse para detectar su estado.

Funcionamiento de una celda de memoria

Lo más común es que la celda tenga tres terminales para transportar señales eléctricas. El terminal de selección, como su nombre lo indica, selecciona la celda para

la operación de escritura o lectura. El terminal de control indica el tipo de operación. Para la escritura, el tercer terminal proporciona la señal que fija el estado de la celda a 1 o a 0. En una lectura, el tercer terminal se usa como salida del estado de la celda.

Organización de la memoria

Elemento básico es la *celda* de memoria, que presenta dos estados estables (que representan el 0 y 1), pueden ser escritas para fijar su contenido, pueden ser leídas para detectar su estado, posee tres terminales para transportar señales eléctricas:

1. de selección: selecciona la celda para la operación de escritura/lectura.
2. de control: indica el tipo de operación.
3. de lectura/escritura: proporciona la señal que fija el estado de la celda a 1 o 0, se utiliza como salida del estado de la celda.

Direccionamiento 2D (Memoria Estática)

Se puede pensar en el direccionamiento 2D como en una lista de celdas apiladas, las cuales tienen un tamaño N (donde N puede ser el tamaño de palabra, por ejemplo). Un tipo de memoria que maneje este tipo de direccionamiento debe tener una conexión con alguna interfaz que le permita conseguir los datos a ingresar, y otra que le indique a qué celda tiene que escribir.

También debe tener una conexión con el lugar a donde tiene que guardar los datos, si es que se realiza una lectura, aunque no siempre es necesario, porque se puede pensar en utilizar la misma conexión para escribir o leer, indicándole a la memoria con un flag qué operación se debe hacer. Esto le agrega complejidad al programador para manejar esta memoria, pero en contrapartida se gana un diseño un poco más simple. Hay que recalcar que cuando se accede, ya sea leyendo o escribiendo, se lee una celda completa (una palabra, por ejemplo).

La problemática que tiene este tipo de disposición de memoria es que es físicamente muy difícil de implementar con una cantidad de registros grandes, generalmente estas memorias oscilan en tamaño entre los 512Kb-1Mb. Esto se debe a que, si pensamos en este tipo de organización como una lista de celdas, esta estructura es muy frágil (pensemos, por ejemplo que si queremos implementar una memoria de 15Mb con palabras de 32bits con celdas que almacenan palabras completas, entonces, tendríamos 3.932.160 celdas de memorias acomodadas una tras de otra; con lo que esta disposición requeriría mucho espacio a lo largo del soporte de la memoria). Hay que mencionar que, si bien puede haber muchas celdas, el tiempo de acceso a ellas con esta disposición es casi constante.

Generalmente, *este tipo de direccionamiento de memoria se aplica en las Memorias Estáticas (Registros de CPU, o Caché), más que nada porque se implementa con Flip-Flops que, a diferencia de los transistores, no necesitan refresco y son más rápidos a la hora de retribuir la información que almacenan.*

Direccionamiento 2D½ (Memoria Dinámica)

Dadas las dificultades a las que nos enfrentamos en el modo de direccionamiento 2D, y teniendo en cuenta el coste (económico, y técnico) que tiene su implementación, se ha optado por otro tipo de modo de direccionamiento que satisfagan requerimientos diferentes a los previamente satisfechos.

De esta manera, surge el modo de direccionamiento 2D½, el cual se organiza de la siguiente manera:

Una memoria que utilice este tipo de direccionamiento, tendría una estructura compuesta por varios chips, las cuales tendrían la responsabilidad de retornar el valor de un bit, el cual se utilizaría para formar el valor de la palabra de N bits que se quiera leer, o escribir en la memoria. Para ello, sería necesario proveerle al chip, la posición de ella que quiero acceder. En consecuencia, cuando a la memoria se le pide acceder a una dirección, la misma se divide en dos partes (a través de decodificadores), una que le indica a la memoria, qué “renglón” se quiere leer (con la cual se ubica a todas los chips necesarios para la reconstrucción de ese “renglón”) y otra, que indica en qué parte de ese chip se debe buscar el bit correspondiente a una fracción del “renglón” a devolver. Una vez identificado cada bit de la palabra, se reconstruye la misma, y se la ubica en la interfaz de salida de la estructura.

El direccionamiento 2D½ mantiene una gran ventaja con respecto al direccionamiento 2D, y esta es que, a pesar de que obtener una palabra de esta memoria requiere más procesamiento, puede extender su capacidad inmensamente con respecto al direccionamiento 2D. De hecho, las memorias RAM que utilizamos en nuestras computadoras tienen capacidades que alcanzan los Gb (más allá del hecho de que cada una de estas plaquetas está compuesta de varios chips, esto habla del tamaño de la estructura que se puede armar con este tipo de direccionamiento).

Ahora bien, a consecuencia de este gran incremento en la capacidad de almacenamiento de una memoria con este modo de direccionamiento, y debido al hecho de que por una cuestión de costes, éstas se construyen con transistores, estas memorias, si bien son mucho más baratas que las 2D, también son considerablemente más lentas. Sobre todo teniendo en cuenta el procesamiento extra que conlleva decodificar una dirección de memoria en particular.

Generalmente, este tipo de direccionamiento de memoria se aplica en las Memorias Dinámicas (DRAM), que se implementan con transistores, los cuales necesitan refrescos periódicos de la información que almacenan.

Direccionamiento 3D (Modelo Teórico)

Este es un modelo teórico, debido a que es, en realidad, un modelo que tiene implementado toda la estructura para hacer funcionar el modelo 2D½. Esto es planteado de forma de que, cada uno de los chips de las que se obtiene un bit, reflejaría una capa del modelo de direccionamiento 3D. Entonces obtendríamos el contenido de una dirección, si obtuviésemos el contenido de cada capa en la misma coordenada X e Y, siendo Z el plano que representa a todas las capas en la misma coordenada.

5.4 Memoria caché

El objetivo de la memoria cache es lograr que la velocidad de la memoria sea lo más rápida posible, consiguiendo al mismo tiempo un tamaño grande al precio de memorias semiconductoras menos costosas.

Hay una memoria principal relativamente grande y más lenta, junto con una memoria cache más pequeña y rápida. La cache contiene una copia de partes de la memoria principal. Cuando el procesador intenta leer una palabra de memoria, se hace una comprobación para determinar si la palabra está en la caché. Si es así, se entrega dicha palabra al procesador. Si no, hay un bloque de memoria principal, consistente en cierto número de palabras, se transfiere a la cache y, después, la palabra es entregada al procesador. Debido al fenómeno de localidad de las referencias, cuando un bloque de datos es captado por la cache para satisfacer una referencia a memoria simple, es probable que se hagan referencias futuras a otras palabras del mismo bloque.

5.5 Memoria externa

Discos magnéticos

Un disco es un plato circular construido con metal o plástico, cubierto por un material magnetizable. Los datos se graban en él y después se recuperan del disco a través de una bobina, llamada cabeza. Durante una operación de lectura o escritura, la cabeza permanece quieta mientras el plato rota bajo ella.

El mecanismo de escritura se basa en el campo magnético producido por el flujo eléctrico, que atraviesa la bobina. Se envían pulsos a la cabeza, y se graban patrones magnéticos en la superficie bajo ella, con patrones diferentes para corrientes positivas y negativas. El mecanismo de lectura se basa en la corriente eléctrica que atraviesa la bobina, producida por un campo magnético que se mueve respecto a la bobina. Cuando la superficie del disco pasa bajo la cabeza, se genera una corriente de la misma polaridad que la que produjo la grabación magnética.

Organización y formato de los datos

Los datos se organizan en un conjunto de anillos concéntricos, llamados pistas. Cada pista es del mismo ancho que la cabeza. Las pistas están separadas por bandas vacías, lo que previene errores debidos a deslineamientos de la cabeza o simplemente interferencias del campo magnético.

Los datos se transfieren hacia y desde el disco en bloques. El bloque es menor que la capacidad de una pista. De acuerdo con esto, los datos se almacenan en regiones del tamaño de un bloque, denominadas sectores.

Características físicas

Las principales características son:

- **Desplazamiento de cabezas:** las cabezas pueden ser:
 - **Fijas:** hay una cabeza de lectura/escritura por pista. Todas las cabezas se montan en un brazo rígido que se extiende a través de todas las pistas.
 - **Móvil:** hay sólo una cabeza de lectura/escritura que se monta en un brazo. Como la cabeza debe poder posicionarse encima de cualquier pista, el brazo debe extenderse o retraerse para este propósito.
- **Transportabilidad del disco:** se divide en dos categorías:
 - **No extraíble:** está permanentemente montado en la unidad de disco.
 - **Extraíble:** puede ser quitado y sustituido por otro disco.
- **Superficies:** puede ser de una sola superficie o doble.
- **Platos:** puede tener un plato único o varios platos apilados verticalmente. En conjunto, se llama paquete de disco y disponen de varios brazos.
- **Mecanismo de la cabeza:** se clasifica en tres tipos:
 - **Separación fija:** la cabeza se posiciona a una distancia fija sobre el plato, dejando entre ambos una capa de aire.
 - **Contacto:** la cabeza efectúa un contacto físico con el medio durante la operación de lectura o escritura. Se usa con los disquetes.
 - **Separación aerodinámica (discos Winchester):** las cabezas están montadas en unidades herméticamente cerradas, casi libres de contaminación. Estos discos operan más cerca de la superficie, por tanto permiten una densidad de datos mayor. La cabeza está en el contorno de una hoja de metal aerodinámica, que reposa suavemente sobre la superficie del plato cuando el disco no se mueve. La presión de aire generada por el giro del disco es suficiente para hacer subir la hoja encima de la superficie.

Parámetros para medir las prestaciones de un disco

Cuando la unidad de disco está funcionando, el disco está rotando a una velocidad constante. Para leer o escribir, la cabeza debe posicionarse en la pista deseada y, al principio del sector deseado de la pista. La selección de la pista implica un movimiento de la cabeza. El tiempo que tarda la cabeza en posicionarse en la pista se conoce como *tiempo de búsqueda* (tiempo inicial de comienzo y tiempo necesario para atravesar las pistas que tienen que cruzarse una vez que el brazo de acceso esté a la velocidad adecuada). El tiempo que tarda el sector en alcanzar la cabeza se llama retardo rotacional, o *latencia rotacional*. La suma del tiempo de búsqueda, si lo hay, y el retardo rotacional se denomina *tiempo de acceso*, o tiempo que se tarda en llegar a la posición de lectura o escritura. Una vez posicionada la cabeza, se lleva a cabo la operación de lectura o escritura, desplazándose el sector bajo la cabeza; esta operación conlleva un *tiempo de transferencia* de datos, que depende de la velocidad de rotación del disco.

Tiempo medio de acceso: Tiempo medio que tarda la aguja en situarse en la pista y el sector deseado; es la suma del Tiempo medio de búsqueda (situarse en la pista), Tiempo de lectura/escritura y la Latencia media (situarse en el sector).

5.6 RAID

Son esquemas estandarizados para el diseño de base de datos para discos múltiples. El esquema RAID consta de seis niveles independientes, desde cero hasta cinco. Estos niveles no implican una relación jerárquica, sino que designan métodos diferentes que poseen tres características comunes:

1. RAID es un conjunto de unidades físicas de disco vistas por el sistema operativo como una única unidad lógica.
2. Los datos se distribuyen a través de las unidades físicas del conjunto de unidades.
3. La capacidad de los discos redundantes se usa para almacenar información de paridad que garantice la recuperación de los datos en caso de fallo de disco.

Los detalles de las características segunda y tercera, cambian según los distintos niveles RAID. RAID 0 no soporta la tercera característica.

La estrategia RAID reemplaza una unidad de disco de gran capacidad por unidades múltiples de menor capacidad, y distribuye los datos de forma que se puedan habilitar accesos simultáneos a los datos de varias unidades, mejorando, por tanto, las prestaciones de E/S, y permitiendo más fácilmente aumentos en la capacidad.

La única contribución de la propuesta RAID es, efectivamente, hacer hincapié en la necesidad de redundancia. El uso de varios dispositivos, además de permitir que varias cabezas y actuadores operen simultáneamente, consiguiendo mayores velocidades de E/S y de transferencia, incrementa la probabilidad de fallo. Para compensar esta disminución de seguridad, RAID utiliza la información de paridad almacenada, que permite la recuperación de datos perdidos debido a un fallo de disco.

Nivel 0 de RAID

No es un verdadero miembro de los RAID porque no incluye redundancia para mejorar las prestaciones.

Para el RAID 0, los datos del usuario y del sistema están distribuidos a lo largo de todos los discos del conjunto. Esto tiene una ventaja frente al uso de un único y gran disco: si hay pendientes dos peticiones diferentes de E/S para dos bloques de datos diferentes, entonces es muy probable que los bloques pedidos estén en diferentes discos. Entonces, las peticiones se pueden emitir en paralelo, reduciendo el tiempo de cola de E/S.

En RAID 0 los datos están organizados en forma de tiras de datos a través de los discos disponibles. Todos los datos del usuario y del sistema se ven como almacenados en un disco lógico. El disco se divide en tiras, que pueden ser bloques, sectores o alguna otra unidad. Un conjunto de tiras consecutivas se llama “franja”.

Nivel 1 de RAID

RAID 1 se diferencia de los niveles 2 a 5 en cómo se consigue la redundancia. En este nivel, la redundancia se logra duplicando todos los datos. Como en RAID 0, se hace una distribución de datos, pero en este caso, cada franja lógica se proyecta en dos discos físicos separados, de forma que cada disco del conjunto tiene un disco espejo que contiene los mismos datos.

La organización RAID 1 tiene tres ventajas:

1. Una petición de lectura puede ser servida por cualquiera de los discos que contienen los datos pedidos, lo que hace que disminuya el tiempo de búsqueda
2. Una petición de escritura requiere que las dos tiras se actualicen y esto puede hacerse en paralelo. Entonces, el resultado de la escritura viene determinada por la menos rápida de las dos escrituras.
3. La recuperación tras un fallo es sencilla

La principal desventaja es el coste. Requiere el doble de espacio del disco lógico que puede soportar.

Las prestaciones de RAID 1 son próximas al doble de las de RAID 0, en cuanto a transferencias de datos y peticiones de E/S.

Nivel 2 de RAID

Los niveles 2 y 3 de RAID usan una técnica de acceso paralelo. En un conjunto de acceso paralelo, todos los discos miembros participan en la ejecución de cada petición de E/S.

Como en los otros esquemas de RAID, se usa la descomposición de datos en tiras. En el caso de RAID 2 y 3, las tiras son muy pequeñas; a menudo, tan pequeñas como un único byte o palabra. Con RAID 2, el código de corrección de errores se calcula a partir de los bits de cada disco, y los bits del código se almacenan en las correspondientes posiciones de bit en varios discos de paridad. Normalmente se usa el código de Hamming, que permite corregir errores en un bit y detectar errores en dos bits.

Aunque RAID 2 requiere menos discos que el nivel 1, es bastante caro. El número de discos redundantes es proporcional al logaritmo del número de discos de datos. En una sola lectura se accede a todos los discos simultáneamente. El controlador del conjunto proporciona los datos perdidos y el código de corrección asociado. Si hay un error en un solo bit, el controlador lo corrige instantáneamente. En una escritura sencilla, la operación de escritura debe acceder a todos los discos de datos y de paridad.

RAID 2 debería ser solamente una elección efectiva en un entorno en el que haya muchos errores de disco. Si hay una alta seguridad en los discos individuales y en las unidades de disco, RAID 2 es excesivo y no se implementa.

Nivel 3 de RAID

Se organiza de manera similar a RAID 2, la diferencia es que requiere sólo un disco redundante. RAID 3 usa un acceso paralelo, con datos distribuidos en pequeñas tiras. En vez de un código de corrección de errores, se calcula un bit de paridad para el conjunto de bits individuales que están en la misma posición en todos los discos de datos.

RAID 3 puede conseguir velocidades de transferencia de datos muy altas y las mejoras de prestaciones en cuanto a E/S es notable. Por otra parte, sólo se puede ejecutar a la vez una petición de E/S, por lo tanto el rendimiento sufre.

Redundancia

En el caso de un fallo en una unidad, se accede a la unidad de paridad y se reconstruyen los datos desde el resto de los dispositivos. Una vez que se sustituye la unidad que ha fallado, los datos que faltan se restauran en la nueva unidad y se reanuda la operación.

La reconstrucción de los datos es bastante sencilla. Consideremos un conjunto de cinco discos, de los que de X0 a X3 contienen datos, y X4 es el disco de paridad. La paridad para el i-ésimo bit se calcula de la siguiente forma:

$$X_{4(i)} = X_{3(i)} \text{ XOR } X_{2(i)} \text{ XOR } X_{1(i)} \text{ XOR } X_{0(i)}$$

Supongamos que la unidad X1 ha fallado. Si sumamos $X_{4(i)} \text{ XOR } X_{1(i)}$ a ambos miembros de la ecuación, tenemos que:

$$X_{4(i)} = Y_{1(i)} \text{ XOR } Y_{2(i)} \text{ XOR } Y_{3(i)} \text{ XOR } Y_{0(i)}$$

Por lo tanto, se puede regenerar el contenido de cualquier tira de datos en X1 a partir del contenido de las correspondientes tiras del resto de los discos del conjunto. Este principio es válido para los niveles 3 a 6 de RAID.

En el caso de que un disco falle, todos los datos estarán todavía disponibles en lo que se denomina modo reducido. En este modo, para lecturas, los datos que faltan se recuperan “al vuelo” con la operación XOR. Cuando se escriben los datos en un conjunto RAID 3 reducido, se debe mantener la consistencia de paridad para regeneraciones posteriores.

Volviendo al funcionamiento global, se requiere que el disco que ha fallado se reemplace y se regenere todo su contenido en el nuevo disco.

Nivel 4 de RAID

Los niveles 4 y 5 de RAID usan una técnica de acceso independiente. En un conjunto de acceso independiente, cada disco opera independientemente, de forma que peticiones de E/S separadas se atienden en paralelo. Debido a esto, son más adecuados los conjuntos de acceso independiente para aplicaciones que requieren velocidades de

petición de E/S altas, y son menos adecuados para aplicaciones que requieren velocidades altas de transferencia de datos.

Como en otros esquemas de RAID, se usan tiras de datos. En RAID 4 y 5 son relativamente grandes. Con RAID 4 se calcula una tira de paridad, bit a bit, a partir de las correspondientes tiras de cada disco de datos, y los bits de paridad se almacenan en la correspondiente tira del disco de paridad.

RAID 4 lleva consigo una penalización en la escritura cuando se realiza una petición de escritura de E/S pequeña. Cada vez que se realiza una escritura, el software de gestión del conjunto debe actualizar, no sólo los datos del usuario, sino también los bits de paridad correspondientes.

Para calcular la nueva paridad, el software de gestión del conjunto debe leer la antigua tira del usuario y la antigua tira de paridad. Entonces, se pueden actualizar las dos con nuevos datos y calcular la nueva paridad.

Cada operación de escritura implica al disco de paridad que se convertirá en un cuello de botella.

Nivel 5 de RAID

RAID 5 está organizado de manera similar al 4. La diferencia es que distribuye las tiras de paridad a lo largo de todos los discos. Una distribución típica es un esquema cíclico. Para un conjunto de n discos, la tira de paridad está en diferentes discos para las primeras n tiras, y este patrón se repite.

La distribución de las tiras de paridad a lo largo de todas las unidades evita el potencial cuello de botella de E/S encontrado en RAID 4.

Nivel 6 de RAID

En este esquema se hacen dos cálculos de paridad distintos, que se almacenan en bloques separados en distintos discos. Por tanto, un conjunto RAID 6 cuyos datos requieran N discos, consta de N+2 discos.

La ventaja de RAID 6 es que proporciona una disponibilidad de los datos extremadamente alta, porque además de la paridad calculada como RAID 4 y 5, tiene un algoritmo de comprobación independiente que hace posible la regeneración de los datos, incluso si dos de los discos fallan. Por otra parte, RAID 6 incurre en una penalización de escritura, ya que cada escritura afecta a dos bloques de paridad.

Organización de las computadoras
Resumen para rendir final – plan 2003

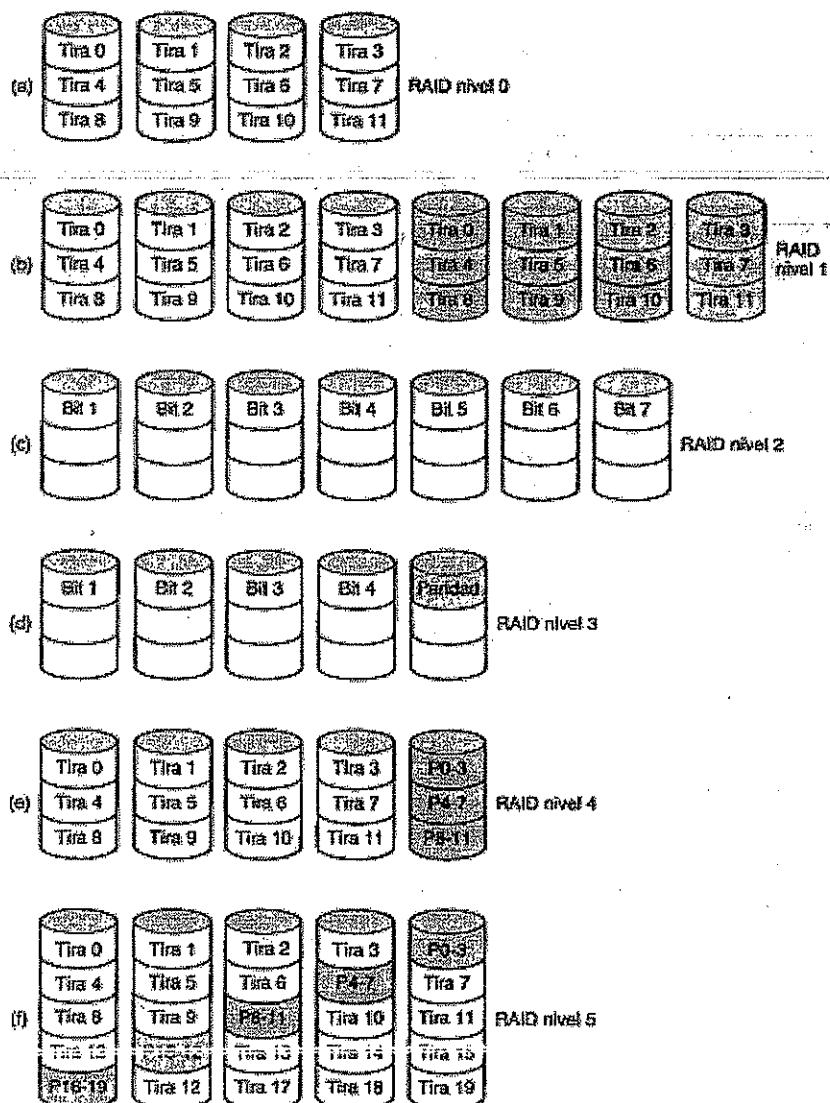
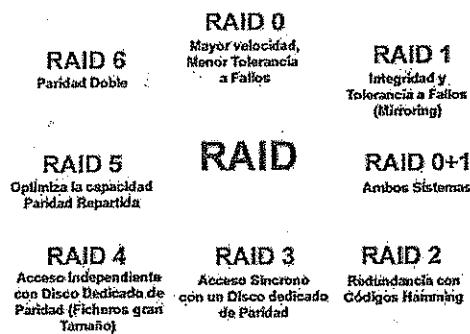


Figura 2-23. RAID niveles 0 a 5. Las unidades de respaldo y paridad se consideran sombreadas.



5.7 Memoria óptica

CD y CD-ROM

Comparten una tecnología similar. Ambos discos se forman a partir de una resina, como un policarbonato, y se cubre con aluminio. La información grabada digitalmente se graba como una serie de hoyos microscópicos en el aluminio con un láser de alta densidad para crear el disco patrón. Este patrón se usa para hacer copias. La superficie con los hoyos de las copias se protege del polvo con una capa de laca transparente.

La información se recupera con un láser de baja potencia situado en un lector. El láser pasa a través de la capa protectora, mientras un motor hace girar el disco. La intensidad de la luz cambia si se encuentra un hoyo. Un fotosensor detecta este cambio, que es convertido en una señal digital.

El disco se divide en una serie de sectores y en una serie de pistas concéntricas. La información se empaqueta con densidad uniforme a lo largo del disco en segmentos del mismo tamaño y se explora a la misma velocidad, rotando el disco a una velocidad variable. El láser, por tanto, lee los hoyos a una velocidad lineal constante.

Comparado con los discos magnéticos tradicionales, tiene tres ventajas principales:

1. La capacidad de almacenamiento es mucho mayor.
2. Se puede replicar en grandes cantidades de forma barata.
3. Es extraíble, permitiendo usar el disco como memoria de archivo.

Las desventajas son:

- Es de sólo lectura
- El tiempo de acceso es mayor que el de una unidad de disco magnético, tanto como medio segundo.

Disco óptico borrable

Se puede escribir y reescribir rápidamente, como cualquier disco magnético. Se usa la tecnología magnético-óptica: se utiliza la energía de un haz de láser, junto con un campo magnético, para grabar y borrar información, invirtiendo los polos magnéticos en una pequeña área del disco cubierta con un material magnético. El haz láser calienta un determinado punto del medio, y un campo magnético puede cambiar la orientación magnética (polo) de este punto, mientras se eleva su temperatura. Como el proceso de polarización no causa un cambio físico en el disco, el proceso se puede repetir varias veces. Para leer, el polo se puede detectar con un haz láser polarizado. La luz polarizada reflejada por un punto particular cambiará su grado de rotación dependiendo de la orientación, norte o sur, del campo magnético.

Las principales ventajas sobre los discos magnéticos son:

- Alta capacidad: más de la mitad
- Intercambiabilidad: son extraíbles
- Seguridad: más vida.

DVD

El DVD es un disco óptico de almacenamiento de datos, cuyo estándar surgió en 1995. Sus siglas corresponden con Digital Versatile Disc en inglés.

Los DVD se dividen en dos categorías: los de capa simple y los de doble capa. Los DVD de capa simple puede guardar hasta 4,7 Gb.

El DVD usa un método de codificación más eficiente en la capa física: los sistemas de detección y corrección de errores utilizados en el CD. El sub-código de CD fue eliminado. Como resultado, el formato DVD es un 47% más eficiente que el CD-ROM, que usa una tercera capa de corrección de errores.

Actualmente, la mayor parte de unidades de CD y DVD tienen una velocidad de rotación constante. La máxima velocidad de transferencia de datos especificada para una cierta unidad y disco se alcanza solamente en los extremos del disco. Por tanto, la velocidad media de la unidad lectora equivale al 50-70% de la velocidad máxima para la unidad y el disco. Aunque esto puede parecer una desventaja, tales unidades tienen un menor tiempo de búsqueda, pues nunca deben cambiar la velocidad de rotación del disco.

Discos magnéticos ópticos (MO)

Usa un láser óptico para aumentar las posibilidades de una unidad de disco magnético convencional. La tecnología de grabación es fundamentalmente magnética. Sin embargo, se usa un láser óptico para focalizar la cabeza grabadora magnética, para poder conseguir una gran capacidad. En este esquema, el disco se recubre con un material, cuya polaridad se puede alterar sólo a altas temperaturas.

La información se escribe en el disco, usando un láser para calentar un pequeño punto de su superficie, y luego aplicando un campo magnético. Cuando el punto se enfriá, adopta la polaridad norte-sur. Este proceso se puede repetir varias veces.

La operación de lectura es puramente óptica. La dirección de la magnetización se puede detectar con un haz de láser polarizado. La luz polarizada reflejada por un punto en concreto cambiará su grado de rotación dependiendo de la orientación del campo magnético.

La principal ventaja frente a los CDs es la longevidad, ya que no se degrada ante repetidas escrituras. Otra ventaja es que es más barato que el almacenamiento magnético.

Cinta magnética

Los sistemas de cinta usan las mismas técnicas de lectura y grabación que los discos. El medio es una cinta de plástico flexible, cubierta por óxido magnético. Como con el disco, los datos se leen y escriben en bloques contiguos, llamados registros físicos de cinta. Los bloques están separados por bandas inter-registros.

Una unidad de cinta es un dispositivo de acceso secuencial, a diferencia del disco que tiene acceso directo.

6. Periféricos

Comunicación hombre-máquina. Comunicación máquina-máquina. Comunicación máquina-mundo físico. Dispositivos de interacción típicos: terminales, pantallas, teclado, mouse, impresora, scanner, tabla digitalizadora, audio. Características de interconexión a cada uno de ellos. Modem.

6.1 Comunicación hombre-máquina

Comunicación hombre-máquina: Permite la comunicación con el usuario de la computadora. Esta comunicación se realiza a través de periféricos (monitor, etc.)

Terminales

Las terminales de las computadoras constan de dos partes: un teclado y un monitor. En el mundo de los mainframes, estas partes a menudo se integran en un solo dispositivo y se conectan a la computadora principal con una línea en serie o telefónica. En el mundo de las computadoras personales son dispositivos independientes. En ambos casos, la tecnología es la misma.

Teclado

Hoy en día los teclados más baratos tienen teclas que se limitan a hacer contacto mecánico cuando se oprimen. Los mejores tienen una lámina de materia elastomérico (una especie de caucho) entre las teclas y la tarjeta de circuitos impresos subyacente. Bajo cada tecla hay un pequeño domo que se aplasta cuando se lo oprime con fuerza suficiente. Un pequeño punto de material conductor dentro del domo cierra el circuito. Algunos teclados tienen un imán debajo de cada tecla que pasa por el interior de una bobina cuando la tecla se oprime e induce una corriente que puede detectarse. Además de éstos, se usan otros métodos, tanto mecánicos como electromagnéticos.

En las computadoras personales, cuando se oprime una tecla, se genera una interrupción que hace que se inicie el manejador de interrupciones del teclado (perteneciente al sistema operativo (SO)). Éste lee un registro del hardware dentro del controlador del teclado para obtener el número de la tecla (1 a 102) que recién se ha oprimido. Cuando se suelta una tecla, se causa la segunda interrupción. El manejo de secuencias de múltiples teclas en las que intervienen SHIFT, CTRL y ALT se realiza totalmente en software.

Monitores de CRT

Un monitor es una caja que contiene un tubo de rayos catódicos (CRT) y sus fuentes de potencia. El CRT contiene un cañón que puede disparar un haz de electrones contra una pantalla fosforescente cerca del frente del tubo (los monitores a color tienen 3 cañones –rojo, verde y azul–). Durante el barrido horizontal, el haz cruza la pantalla describiendo una línea horizontal de izquierda a derecha y de arriba abajo en ella. Luego regresa al borde izquierdo e inicia el siguiente barrido. Este tipo de dispositivo se llama de *barrido por cuadro*. Una vez terminado de barrer, se produce, mediante un voltaje más lento que el horizontal, un barrido vertical, que regresa el haz otra vez en el extremo superior izquierdo. Una imagen de pantalla completa normalmente se redibuja entre 30 y 60 veces cada segundo.

Para producir un patrón de puntos en la pantalla, hay una rejilla dentro del CRT que, cuando se aplica en ella un voltaje positivo el haz choca la pantalla y brilla y, cuando el voltaje es negativo la pantalla no brilla. Así, el voltaje aplicado a la rejilla hace que el patrón de bits aparezca en pantalla. Este mecanismo permite convertir una señal eléctrica binaria en una imagen formada por puntos brillantes y oscuros.

LCD (pantalla de cristal líquido)

Los cristales líquidos son moléculas orgánicas viscosas que fluyen como un líquido pero también tienen una estructura espacial, como un cristal. Cuando todas las moléculas están alineadas en la misma dirección, las propiedades ópticas del cristal dependen de la dirección y la polarización de la luz incidente. Con la ayuda de un campo eléctrico es posible modificar la alineación molecular y por ende las propiedades ópticas del cristal. En particular, si se coloca una lámpara detrás de un cristal líquido, es posible controlar eléctricamente la intensidad de la luz que se percibe a través del cristal.

Una pantalla LCD consiste en dos placas de vidrio paralelas entre las que hay un volumen sellado que contiene cristal líquido. Cada placa tiene conectados electrodos transparentes que crean campos eléctricos en el cristal. Una luz ilumina la pantalla desde atrás. Diferentes partes de la pantalla reciben diferentes voltajes, y con esto se controla la imagen que se exhibe. En ausencia de campo eléctrico, la pantalla es uniformemente brillante, donde se aplica voltaje, la luz no atraviesa. Pegados a la pantalla hay filtros polarizantes que permiten o no pasar la luz.

En los LCD color se utilizan filtros ópticos para separar la luz blanca en componentes rojo, verde y azul.

Terminales de mapa de caracteres

Se usan comúnmente tres tipos de terminales: de mapa de caracteres, de mapa de bits y RS-232-C. Todas ellas pueden usar cualquier tipo de teclado, pero difieren en la forma en que se comunican con ellas y en el manejo de las salidas.

En una computadora personal hay dos formas de organizar las salidas que se envían a la pantalla: un mapa de caracteres y un mapa de bits. En el mapa de caracteres, en la tarjeta de comunicaciones en serie hay una porción de memoria, llamada *memoria de video*, además de algunos circuitos para acceder al bus y generar señales de video.

Para exhibir caracteres, la CPU los copia en la memoria de video en bytes alternados. Cada carácter tiene asociado un byte de atributo que describe la forma en que debe exhibirse el carácter.

La tarea de la tarjeta de video consiste en traer una y otra vez caracteres de la RAM de video y generar la señal (analógica) necesaria que se alimenta al monitor.

Terminales de mapa de bits

Usan píxeles que representan un bit de información. La RAM de video se ve como un gran arreglo de bits. El software puede crear ahí cualquier patrón que desee, y se exhibirá instantáneamente.

Las terminales de mapa de bits suelen usarse para presentar imágenes que contienen varias ventanas. Usar mapa de bits tiene dos desventajas: primera, requieren una cantidad considerable de RAM de video. Para obtener color fiel, se requieren 8 bits para cada uno de los colores primarios, o sea 3bytes/píxel. Así, una pantalla de 1024x768 requiere 2.3Mb de RAM de video. La segunda desventaja es su desempeño, ya que es muy costoso desplazar la imagen en la pantalla.

Terminales RS-232-C

Se crearon con el fin de que cualquier terminal pueda usarse con (casi) cualquier computadora. Tienen un conector estandarizado de 25 terminales. Cuando esta Terminal se encuentra muy separada de la computadora, se usa un módem para conectarlas. Para comunicarse, la computadora y la terminal contienen un chip llamado *transmisor-receptor universal asincrónico* (UART, un convertidor de paralelo a serie), además de lógica para acceder al bus. Para exhibir un carácter, la computadora lo trae de su memoria principal y lo presenta al UART, que entonces lo mete en el cable RS-232-C bit por bit. En la Terminal, otro UART recibe los bits y reconstruye el carácter, que luego se exhibe en la pantalla. Las entradas del teclado de la Terminal se someten a una conversión paralelo-serie en la Terminal y el UART las reensambla en la computadora.

Ratones

Un *mouse* es una pequeña caja de plástico que descansa sobre la mesa junto al teclado. Cuando el ratón se mueve sobre la mesa, también se mueve un puntero en la

pantalla, que apunta a los elementos. El ratón tiene uno, dos o tres botones en la parte superior, que permiten seleccionar opciones de menú.

Existen tres tipos de ratones: mecánicos, ópticos y optomecánicos.

1. **Mecánicos:** tenían dos ruedas de caucho que sobresalían por debajo y tenían sus ejes perpendiculares entre sí. Cuando el ratón se movía paralelo a su eje principal, una rueda giraba. Cuando se movía perpendicularmente, giraba la otra rueda. Cada rueda impulsaba un resistor variable. Si se medían los cambios en la resistencia, era posible ver qué tanto había girado cada rueda y así calcular qué tanto se había movido el ratón en cada dirección. Este tipo se reemplazó por uno con una esfera que sale ligeramente de su base.
2. **Ópticos:** no tiene rueda ni esfera, tiene un LED y un fotodetector en la base. El ratón se usa sobre una base que contiene una cuadrícula de líneas. A medida que el ratón se mueve sobre ella, el fotodetector percibe el paso de las líneas por los cambios en la cantidad de luz que se refleja. Unos circuitos internos cuentan el número de líneas que se cruzan en cada dirección.
3. **Optomecánico:** tiene una esfera que hace girar dos ejes perpendiculares entre sí. Los ejes se conectan a codificadores provistos de ranuras a través de las cuales puede pasar la luz. A medida que el ratón se mueve, los ejes giran y pulsos de luz inciden sobre los detectores cada vez que una ranura pasa entre un LED y su detector. El número de pulsos detectados es proporcional a la cantidad de movimiento.

La configuración más común de un mouse es hacer que envíe una secuencia de 3 bytes a la computadora cada vez que se mueva cierta distancia mínima, llamada mickey. Los bytes llegan por una línea serial, bit por bit. El primer byte contiene un entero con signo que indica cuántas unidades se movió el ratón en la dirección x en los últimos 100ms. El segundo byte da la misma información para la dirección y. El tercer byte contiene la situación actual de los botones.

Software de bajo nivel acepta esta configuración y convierte los movimientos relativos en una posición absoluta. Luego exhibe una flecha en la pantalla en la posición correspondiente.

Tabla digitalizadora

Está formada por un dispositivo apuntador móvil más un panel sensible de hasta $1m^2$ de tamaño. La resolución puede llegar hasta milésimas de milímetro. El dispositivo apuntador suele ser una mirilla o lápiz electrónico similar al lápiz óptico que tiene cuatro botones que sirven, entre otras funciones para recoger las coordenadas de la posición actual.

Para determinar la posición del puntero se pueden diferenciar tres métodos.

- **Electromagnético:** debajo de la tableta existe una matriz de hilos muy densa. El procesador de la tableta envía pulsos eléctricos a través de la rejilla, recorriendo todas las columnas para cada una de las filas. Cuando coinciden los pulsos de las X's y de las Y's se induce un voltaje en el puntero que sirve para determinar su

posición en función del instante de tiempo en que se detectó. El espacio de la rejilla puede variar entre 0,25 y 10mm.

- Por ultrasonidos: el puntero emite ultrasonidos que son recogidos por micrófonos.
- Por presión: existe una rejilla al igual que en los teclados de membrana.

6.2 Comunicación máquina-mundo físico

Permite la comunicación con dispositivos remotos. En muchas aplicaciones, el computador se encuentra conectado directamente a un sistema físico exterior, sobre el que realiza funciones tales como adquisición de datos, control de procesos, control de instrumentación de laboratorios, etc. Para realizar esta conexión, se emplean una serie de periféricos que se denominan de forma genérica periféricos de control.

Existen cuatro tipos de periféricos de control: Cadena de lectura analógica, Cadena de lectura digital, Cadena de acción analógica, Cadena de acción digital.

La conexión con el módulo de E/S se realiza a través de señales de control, estado y datos. Los datos se intercambian en forma de un conjunto de bits que son enviados/recibidos a/desde el módulo de E/S. Las señales de control determinan la función que debe realizar el dispositivo. Las de estado indican el estado del chip.

Puertos normalizados

Permiten la conexión desde el exterior hacia la placa base. Funcionan con las características estándar y su fabricante tiene que conocer las normas estándar para que no surjan problemas de conexión con los periféricos. Las conexiones o puertos son:

- Los **puertos serie (COM)**. Transmiten la información BIT a BIT. No son adecuados para transferir grandes cantidades de información. En ellos se conectan el módem, cuando es externo.
- Los **puertos paralelos (LPT)**. Permiten una mayor velocidad en la transmisión de datos. La información se transmite en bytes o múltiplos. Es la conexión típica de impresoras o escáneres.
- Los **puertos USB (Universal Serial Bus)**. En ellos se conectan, en general, dispositivos que necesitan una alta velocidad de transferencia de datos como, por ejemplo, un escáner o una cámara digital, aunque también existen impresoras con este tipo de conexión.

Impresoras monocromáticas

Impresora de matriz

El tipo de impresora más económico es la **impresora de matriz**, en la que una cabeza de impresión que contiene entre 7 y 24 agujas activables electromagnéticamente se mueve a lo largo de cada línea de impresión. Las de 7 agujas imprimen 80 caracteres en una matriz de 5x7 a lo largo de la línea. De hecho, la línea de impresión consiste en 7 líneas horizontales, cada una de las cuales consta de $5 \times 80 = 400$ puntos. Cada punto

puede imprimirse o no, dependiendo de los caracteres que han sido seleccionados para tal efecto. Las impresoras de matriz son económicas y muy confiables, pero son lentas, ruidosas y malas para imprimir gráficos. Se usan en tickets, o para imprimir en formatos grandes, entre otros.

Impresora de inyección de tinta

Para la impresión casera de bajo costo, las *impresoras de inyección de tinta* son las favoritas. La cabeza de impresión móvil, que lleva un cartucho de tinta, se mueve horizontalmente a lo ancho del papel mientras rocía tinta con sus diminutas boquillas. Dentro de cada boquilla, una pequeña gota de tinta se calienta eléctricamente más allá de su punto de ebullición, hasta que hace explosión, sale y choca con el papel. Luego la boquilla se enfriá y el vacío que se produce succiona otra gota de tinta. Las impresoras de inyección suelen tener definiciones de 300dpi (puntos por pulgada) a 1440dpi. Son económicas, silenciosas y con buena calidad, pero también son lentas, sus cartuchos son caros y producen impresiones saturadas de tinta.

Impresora láser

La impresora láser combina una alta calidad, excelente flexibilidad, buena velocidad y costo moderado en un mismo periférico. El corazón de la impresora es un cilindro de precisión giratorio. Al principio de cada ciclo de página, el cilindro se carga hasta cerca de 1000 volts y se recubre con un material fotosensible. Luego la luz de un láser se mueve a lo largo del cilindro de forma muy similar a como un haz de electrones se mueve a través de un CRT, sólo que en lugar de efectuar la desviación horizontal con una aplicación de un voltaje, se usa un espejo octogonal giratorio.

Una vez que se ha pintado una línea de puntos, el cilindro gira una fracción de grado para que pueda pintarse la siguiente línea. En algún momento, la primera línea de puntos llega al tóner, un depósito de polvo negro sensible a las cargas electrostáticas. El tóner es atraído hacia los puntos que todavía conservan su carga, y así forman una imagen visual de esa línea. Un poco más adelante en el trayecto de transporte, el cilindro recubierto con tóner se opone contra el papel y transfiere a éste el polvo negro. Luego el papel pasa entre rodillos calientes que fusionan el tóner con el papel de forma permanente, lo que fija la imagen. Al continuar su rotación, el cilindro pierde su carga y un raspador elimina cualquier residuo de tóner, antes de que el cilindro se vuelva a cargar y recubrir para la siguiente página.

Las impresoras láser imprimen páginas completas, con una resolución de 300 a 1200dpi.

Impresoras a color

Las imágenes a color pueden verse de dos maneras: como luz transmitida o como luz reflejada. Las primeras, como las que se producen en los CRT, se forman a partir de la superposición lineal de los tres colores primarios aditivos, rojo, verde y azul. Las de luz reflejada, como las fotografías, absorben ciertas longitudes de onda de la luz y reflejan el resto. Estas imágenes se forman por la superposición lineal de los tres

colores primarios sustractivos, cian (absorbe el rojo), amarillo (absorbe el azul) y magenta (absorbe el verde). Por esta razón, los sistemas de impresión a color emplean cuatro tintas, CYMK (Cian, Yellow, Magenta y black), aunque los monitores emplean RGB.

El conjunto total de colores que puede producir es su gama. En los mejores casos, cada color viene con 256 intensidades, lo que da aproximadamente 16 millones de colores discretos.

Scanner

Se utiliza para convertir, mediante el uso de la luz, imágenes impresas o documentos a formato digital. Al obtenerse una imagen digital se puede corregir defectos, recortar un área específica de la imagen o también digitalizar texto. Hay varios tipos. Hoy en día los más extendidos son los planos.

- *De rodillo.* Como el escáner de un fax
- "de barras". Se usan para registrar código de barras
- *De mano.* En su momento muy económicos, pero de muy baja calidad. Prácticamente extintos.
- *Planos:* también llamados escáneres de sobremesa, están formados por una superficie plana de vidrio sobre la que se sitúa el documento a escanear, generalmente opaco, bajo la cual un brazo se desplaza a lo largo del área de captura. Montados en este brazo móvil se encuentran la fuente de luz y el fotosensor de luz. Conforme va desplazándose el brazo, la fuente de luz baña la cara interna del documento, recogiendo el sensor los rayos reflejados, que son enviados al software de conversión analógico/digital para su transformación en una imagen de mapa de bits, creada mediante la información de color recogida para cada píxel. La mayoría de estos escáneres pueden trabajar en escala de grises (256 tonos de gris) y en color (24 y 32 bits) y por lo general tienen un área de lectura de dimensiones 22 x 28 cm. y una resolución real de escaneado de entre 300 y 2400 ppp. Los escáneres planos son los más accesibles y usados, pues son veloces, fáciles de manejar, producen imágenes digitalizadas de calidad aceptable (sobre todo si están destinadas a la web) y son bastante baratos. La mayor desventaja de estos escáneres es la limitación respecto al tamaño del documento a escanear, que queda limitado a los formatos A5 o A4.
- *Cenitales:* es un tipo de escáner que se utiliza para hacer copias digitales de libros o documentos que, por ser viejos o extremadamente valiosos, para que no se deterioren escaneándolos con otro tipo de escáner.

Estos escáneres consisten en una cámara montada en un brazo que toma fotos del elemento deseado. Su ventaja principal es que los libros no tienen que ser abiertos completamente (como pasa en la mayoría de los escáneres planos). El escaneo de volúmenes encuadrados se realiza gracias a que la fuente de luz y el sensor se encuentran ensamblados a un brazo de trayectoria aérea.

- *De tambor:* Los escáneres de tambor son los que más fielmente reproducen el documento original, ya que producen digitalizaciones de gran calidad y resolución (hasta 16.000 ppp de resolución óptica). Sus problemas son la velocidad de escaneo (son lentos), no son indicados para documentos de papel

quebradizo porque es necesario curvarlo sobre el tambor y requieren un alto nivel de habilidad por parte del operador. Además, son bastante caros.

Los originales, normalmente transparencias (aunque se pueden escanear opacos también), se colocan en un cilindro transparente de cristal de gran pureza, que a su vez se monta en el escáner. El tambor gira entonces a gran velocidad mientras se hace la lectura de cada punto de la imagen. La fuente de luz suele ser un láser que se encuentra dentro del tambor (para transparencias) o fuera (para opacos), y el sensor es un tubo fotomultiplicador (PMT) que recibe la luz de un único punto de la imagen en cada instante.

Producen digitalizaciones de alta resolución y buena gama dinámica entre bajas y altas luces, con imágenes en colores primarios, que pueden ser convertidas en CMYK mientras el lector recorre la imagen.

- *Otros tipos.* Existen tipos de escáneres especializados en un trabajo determinado (por ejemplo para escanear microfilms, o para obtener el texto de un libro completo, para negativos...)

Antes los escáneres usaban conexiones paralelas que no podían ir más rápido de los 70 kilobytes/segundo, SCSI-II se adoptó para los modelos profesionales y aunque era algo más rápido (unos cuantos megabytes por segundo) era bastante más caro.

Hoy en día los modelos más recientes vienen equipados con conexión USB, que posee una tasa de transferencia de 1.5 megapixel por segundo para los USB 1.1 y de hasta 60 megapixel por segundo para las conexiones USB 2.0, lo que elimina en gran medida el cuello de botella que se tenía al principio.

Funcionamiento

Todos los scanner utilizan el mismo principio de reflexión o de transmisión de la luz. La imagen se coloca debajo del cabezal lector, que consiste en una fuente de luz y un sensor que mide la cantidad de luz reflejada o transmitida. Después la información analógica se convierte en digital, utilizando un conversor Analógico/Digital (A/D).

El sensor de luz suele ser un dispositivo CCD (*Charge Coupled Device*) que convierte la intensidad de la luz recibida en un voltaje proporcional. El cabezal monta una tira con miles de estos elementos. El scanner emite luz sobre tres filtros (rojo, verde y azul) y la luz reflejada en el documento a digitalizar es dirigida hacia la tira de sensores mediante un sistema de espejos y lentes. El CCD actúa como un fotómetro que produce un voltaje que posteriormente se convierte en información digital.

6.3 Comunicación máquina-máquina

Permite la comunicación con elementos de equipo (discos magnéticos, cintas, etc.). La comunicación es digital. Si los computadores están cerca se conectan directamente en Banda Base, mediante un bus paralelo que transmite bytes o palabras. Si existe una distancia de más de 10 o 100 mts. se utilizan los módems.

Módems

Sirven para la comunicación entre máquinas usando la línea telefónica. Una línea telefónica no es apropiada para transmitir señales de computadora, que generalmente representan un 0 como 0 volts y un 1 como 3 a 5 volts. Las señales de dos niveles sufren una distorsión cuando se transmiten por línea telefónica de grado de voz y produce errores, por lo tanto se usa una señal de onda senoidal, cuya frecuencia (entre 1000 y 2000 Hz) se llama *portadora*. Las pulsaciones de una onda senoidal son predecibles. Si se varía la amplitud, la frecuencia o fase se puede transmitir una secuencia de unos y ceros. Este proceso se llama *modulación*. En la *modulación de amplitud* se usan dos niveles de voltajes distintos para el 1 y para el 0. Cuando se transmite un 1 se escucha un ruido fuerte y un silencio cuando se transmite un 0.

En la *modulación de frecuencia* la frecuencia portadora es diferente para el 0 y para el 1. Se escucharán dos tonos, uno para cada uno.

En la *modulación de fase simple*, la amplitud y la frecuencia no cambian, pero la fase portadora se invierte 180° cuando los datos cambian de 0 a 1 o de 1 a 0. En los sistemas más avanzados, al principio de cada intervalo de tiempo indivisible, la fase portadora se desplaza abruptamente 42, 135, 225 o 315 grados, lo que permite transmitir dos bits en cada intervalo de tiempo. Esto se conoce como codificación por fase *dibit*. El número de intervalos se llama *tasa de Bauds*. La tasa máxima para una transmisión telefónica es de 2400.

Los módems modernos operan con tasas de datos de entre 28,800 bits por segundo (bps) y 57,600bps y normalmente a tasas de bauds mucho más bajas. Emplean una combinación de técnicas para enviar varios bits por baud, modulando la amplitud, la frecuencia y la fase. Casi todos ellos son *díplex*, o sea que pueden transmitir en ambas direcciones a la vez. Los que sólo transmiten en una dirección al mismo tiempo se llaman *semidíplex* y los que sólo transmiten en una dirección son *simplex*.

Tarjeta de sonido

Es una tarjeta de expansión para computadoras que permite la entrada y salida de audio bajo el control de un driver. Una tarjeta de sonido típica, incorpora un chip de sonido, que por lo general contiene el conversor digital-analógico, el cual cumple con la importante función de traducir formas de ondas grabadas o generadas digitalmente en una señal analógica y viceversa. Esta señal es enviada a un conector, en donde se puede conectar cualquier otro dispositivo como un amplificador, un altavoz, etc. Para poder grabar y reproducir audio al mismo tiempo con la tarjeta de sonido debe poseer la característica *full-duplex* para que los dos conversores trabajen de forma independiente.

