PROGRAMACION DE SISTEMAS

TRADUCTORES DE BAJO NIVEL

CATEDRATICO:LIC. ENRIQUE MARTINEZ GUTIERREZ

ALUMNOS:

RAMÍREZ RESENDIZ BALDEMAR
RUÍZ MORALES ROBERTO C.
MARTÍNEZ RAMÍREZ PEDRO F.
GUZMÁN GUTIÉRREZ PEDRO A. - peter_g_g66@hotmail.com

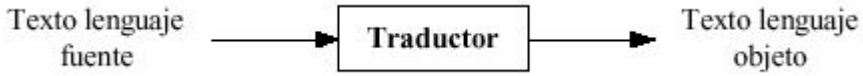
GRADO Y GRUPO: 4º "A"

TAPACHULA DE CORDOVA Y ORDOÑEZ, CHIAPAS A 8 DE SEPTIEMBRE DE 2006

Unidad 1:TRADUCTORES DE BAJO NIVEL

¿QUÉ ES UN TRADUCTOR?

Un traductor es cualquier programa que toma como entrada un texto escrito en un lenguaje, llamado fuente y da como salida otro texto en un lenguaje, denominado objeto.



En el caso de que el lenguaje fuente sea un lenguaje de programación de alto nivel y el objeto sea un lenguaje de bajo nivel (ensamblador o código de máquina), a dicho traductor se le denomina compilador. Un ensamblador es un compilador cuyo lenguaje fuente es el lenguaje ensamblador. Un intérprete no genera un programa equivalente, sino que toma una sentencia del programa fuente en un lenguaje de alto nivel y la traduce al código equivalente y al mismo tiempo lo ejecuta. Históricamente, con la escasez de memoria de los primeros ordenadores, se puso de moda el uso de intérpretes frente a los compiladores, pues el programa fuente sin traducir y el intérprete juntos daban una ocupación de memoria menor que la resultante de los compiladores. Por ello los primeros ordenadores personales iban siempre acompañados de un intérprete de BASIC (Spectrum, Commodore VIC-20, PC XT de IBM, etc.). La mejor información sobre los errores por parte del compilador así como una mayor velocidad de ejecución del código resultante hizo que poco a poco se impusieran los compiladores. Hoy en día, y con el problema de la memoria prácticamente resuelto, se puede hablar de un gran predominio de los compiladores frente a los intérpretes, aunque intérpretes como los incluidos en los navegadores de Internet para interpretar el código JVM de Java son la gran excepción.

Ventajas de compilar frente a interpretar:

Se compila una vez, se ejecuta n veces.

En bucles, la compilación genera código equivalente al bucle, pero interpretándolo se traduce tantas veces una línea como veces se repite el bucle.

El compilador tiene una visión global del programa, por lo que la informació

Ventajas del intérprete frente al compilador:

Un intérprete necesita menos memoria que un compilador. En principio eran más abundantes dado que los ordenadores tenían poca memoria.

Permiten una mayor interactividad con el código en tiempo de desarrollo.

Un compilador no es un programa que funciona de manera aislada, sino que necesita de otros programas para conseguir su objetivo: obtener un programa ejecutable a partir de un programa fuente en un lenguaje de alto nivel. Algunos de esos programas son el preprocesador, el linker, el depurador y el ensamblador. El preprocesador se ocupa (dependiendo del lenguaje) de incluir ficheros, expandir macros, eliminar comentarios, y otras tareas similares. El linker se encarga de construir el fichero ejecutable añadiendo al fichero objeto generado por el compilador las cabeceras necesarias y las funciones de librería utilizadas por el programa fuente. El depurador permite, si el compilador ha generado adecuadamente el programa objeto, seguir paso a paso la ejecución de un programa. Finalmente, muchos compiladores, en vez de generar código objeto, generan un programa en lenguaje ensamblador que debe después convertirse en un ejecutable mediante un programa ensamblador.

TRADUCTORES DE BAJO NIVEL

SB1 Ensambladores. Función de un ensamblador. Estructuras de información necesarias. Procedimientos específicos para el proceso de ensamble. Pasadas del texto fuente. Tablas previamente construidas y tablas generadas en el ensamble. Tratamiento de operandos y modos de direccionamiento de la máquina objeto. Ensambladores residentes y cruzados. Ensamble condicional y su relación con el macroensamble.
SB2 Macroensambladores. Definición de macroinstrucciones en ensamblador. Expansión de macroinstrucciones. Macroinstrucciones con parámetros. Ensamble condicional. Macrollamadas. Tablas y pasos del macroensamblador. Relación con el ensamblador.
SB3 Macroprocesadores. Macros: definición y expansión simple, paramétrica, con anidamiento, etc. Tablas asociadas. Bibliotecas de macros. Expansión condicional. Variables y operadores del proceso de expansión. Pasadas del texto fuente (preprocesamiento).

VENTAJAS Y DESVENTAJAS DEL LENGUAJE ENSAMBLADOR

Una vez que hemos visto la evolución de los lenguajes, cabe preguntarse: ¿En estos tiempos "modernos", para qué quiero el Lenguaje Ensamblador?

El proceso de evolución trajo consigo algunas desventajas, que ahora veremos como las ventajas de usar el Lenguaje Ensamblador, respecto a un lenguaje de alto nivel:

- Velocidad
- Eficiencia de tamaño
- Flexibilidad
- Por otro lado, al ser un lenguaje más primitivo, el Ensamblador tiene ciertas desventajas respecto a los lenguajes de alto nivel:
- Tiempo de programación
- Programas fuente grandes
- Peligro de afectar recursos inesperadamente
- Falta de portabilidad
- Velocidad

El proceso de traducción que realizan los intérpretes, implica un proceso de cómputo adicional al que el programador quiere realizar. Por ello, nos encontraremos con que un intérprete es siempre más lento que realizar la misma acción en Lenguaje Ensamblador, simplemente porque tiene el costo adicional de estar traduciendo el programa, cada vez que lo ejecutamos.
De ahí nacieron los compiladores, que son mucho más rápidos que los intérpretes, pues hacen la traducción una vez y dejan el código objeto, que ya es Lenguaje de Máquina, y se puede ejecutar muy rápidamente. Aunque el proceso de traducción es más complejo y costoso que el de ensamblar un programa, normalmente podemos despreciarlo, contra las ventajas de codificar el programa más rápidamente.

Sin embargo, la mayor parte de las veces, el código generado por un compilador es menos eficiente que el código equivalente que un programador escribiría. La razón es que el compilador no tiene tanta inteligencia, y requiere ser capaz de crear código genérico, que sirva tanto para un programa como para otro; en cambio, un programador humano puede aprovechar las características específicas del problema, reduciendo la generalidad pero al mismo tiempo, no desperdicia ninguna instrucción, no hace ningún proceso que no sea necesario.
Para darnos una idea, en una PC, y suponiendo que todos son buenos programadores, un programa para ordenar una lista tardará cerca de 20 veces más en Visual Basic (un intérprete), y 2 veces más en C (un compilador), que el equivalente en Ensamblador. Por ello, cuando es crítica la velocidad del programa, Ensamblador se vuelve un candidato lógico como lenguaje.

Ahora bien, esto no es un absoluto; un programa bien hecho en C puede ser muchas veces más rápido que un programa mal hecho en Ensamblador; sigue siendo sumamente importante la elección apropiada de algoritmos y estructuras de datos. Por ello, se recomienda buscar optimizar primero estos aspectos, en el lenguaje que se desee, y solamente usar Ensamblador cuando se requiere más optimización y no se puede lograr por estos medios.

Tamaño

Por las mismas razones que vimos en el aspecto de velocidad, los compiladores e intérpretes generan más código máquina del necesario; por ello, el programa ejecutable crece. Así, cuando es importante reducir el tamaño del ejecutable, mejorando el uso de la memoria y teniendo también beneficios en velocidad, puede convenir usar el lenguaje Ensamblador. Entre los programas que es crítico el uso mínimo de memoria, tenemos a los virus y manejadores de dispositivos (drivers). Muchos de ellos, por supuesto, están escritos en lenguaje Ensamblador.

Flexibilidad

Las razones anteriores son cuestión de grado: podemos hacer las cosas en otro lenguaje, pero queremos hacerlas más eficientemente. Pero todos los lenguajes de alto nivel tienen limitantes en el control; al hacer abstracciones, limitan su propia capacidad. Es decir, existen tareas que la máquina puede hacer, pero que un lenguaje de alto nivel no permite. Por ejemplo, en Visual Basic no es posible cambiar la resolución del monitor a medio programa; es una limitante, impuesta por la abstracción del GUI Windows. En cambio, en ensamblador es sumamente sencillo, pues tenemos el acceso directo al hardware del monitor.

Resumiendo, la flexibilidad consiste en reconocer el hecho de que

Todo lo que puede hacerse con una máquina, puede hacerse en el lenguaje ensamblador de esta máquina; los lenguajes de alto nivel tienen en una u otra forma limitantes para explotar al máximo los recursos de la máquina.

Tiempo de programación

Al ser de bajo nivel, el Lenguaje Ensamblador requiere más instrucciones para realizar el mismo proceso, en comparación con un lenguaje de alto nivel. Por otro lado, requiere de más cuidado por parte del programador, pues es propenso a que los errores de lógica se reflejen más fuertemente en la ejecución.

Por todo esto, es más lento el desarrollo de programas comparables en Lenguaje Ensamblador que en un lenguaje de alto nivel, pues el programador goza de una menor abstracción.

Programas fuente grandes

Por las mismas razones que aumenta el tiempo, crecen los programas fuentes; simplemente, requerimos más instrucciones primitivas para describir procesos equivalentes. Esto es una desventaja porque dificulta el mantenimiento de los programas, y nuevamente reduce la productividad de los programadores.

Peligro de afectar recursos inesperadamente

Tenemos la ventaja de que todo lo que se puede hacer en la máquina, se puede hacer con el Lenguaje Ensamblador (flexibilidad). El problema es que todo error que podamos cometer, o todo riesgo que podamos tener, podemos tenerlo también en este Lenguaje. Dicho de otra forma, tener mucho poder es útil pero también es peligroso.

En la vida práctica, afortunadamente no ocurre mucho; sin embargo, al programar en este lenguaje verán que es mucho más común que la máquina se "cuelgue", "bloquee" o "se le vaya el avión"; y que se reinicialize. ¿Por qué?, porque con este lenguaje es perfectamente posible (y sencillo) realizar secuencias de instrucciones inválidas, que normalmente no aparecen al usar un lenguaje de alto nivel.

En ciertos casos extremos, puede llegarse a sobreescribir información del CMOS de la máquina (no he visto efectos más riesgosos); pero, si no la conservamos, esto puede causar que dejemos de "ver" el disco duro, junto con toda su información.

Falta de portabilidad

Como ya se mencionó, existe un lenguaje ensamblador para cada máquina; por ello, evidentemente no es una selección apropiada de lenguaje cuando deseamos codificar en una máquina y luego llevar los programas a otros sistemas operativos o modelos de computadoras. Si bien esto es un problema general a todos los lenguajes, es mucho más notorio en ensamblador: yo puedo reutilizar un 90% o más del código que desarrollo en "C", en una PC, al llevarlo a una RS/6000 con UNIX, y lo mismo si después lo llevo a una Macintosh, siempre y cuando esté bien hecho y siga los estándares de "C", y los principios de la programación estructurada. En cambio, si escribimos el programa en Ensamblador de la PC, por bien que lo desarrollemos y muchos estándares que sigamos, tendremos prácticamente que reescribir el 100 % del código al llevarlo a UNIX, y otra vez lo mismo al llevarlo a Mac.

CPU Y SUS COMPONENTES PRINCIPALES

Un elemento importante del hardware de la PC es la unidad del sistema, que contiene una tarjeta de sistema, fuente de poder y ranuras de expansión para tarjetas opcionales. Los elementos de la tarjeta de sistema son un microprocesador, memoria de solo lectura (ROM) y memoria de acceso aleatorio (RAM).

El cerebro de la PC y compatibles es un microprocesador basado en la familia 8086 de Intel, que realiza todo el procesamiento de datos e instrucciones. Los procesadores varían en velocidad y capacidad de memoria, registros y bus de datos. Un bus de datos transfiere datos entre el procesador, la memoria y los dispositivos externos.

Aunque existen muchos tipos de computadoras digitales según se tenga en cuenta su tamaño, velocidad de proceso, complejidad de diseño físico, etc., los principios fundamentales básicos de funcionamiento son esencialmente los mismos en todos ellos.

Se puede decir que una computadora está formada por tres partes fundamentales, aunque una de ellas es subdividida en dos partes no menos importantes. En la figura 2.1 se muestran dichas partes, llamadas genéricamente unidades funcionales debido a que, desde el punto de vista del funcionamiento, son independientes.

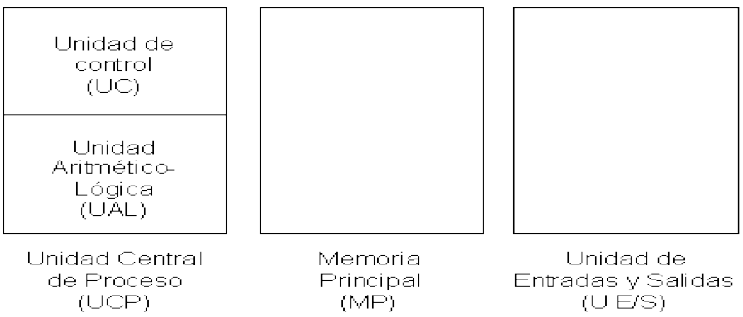


Figura 2.1. Unidades funcionales de una computadora.

El nombre de cada parte nos indica la función que realiza. Así, la Unidad Central de Proceso (CPU) es la que coordina el funcionamiento conjunto de las demás unidades y realiza los cálculos necesarios; por eso la podemos subdividir en una Unidad de Control (UC) y en una unidad de cálculo o Unidad Aritmético-Lógica (UAL).

La unidad de Memoria Principal (MP) se encarga de almacenar las instrucciones que realizará la Unidad de Control al ejecutar un programa y los datos que serán procesados. La Unidad de Entradas y Salidas será la encargada de la comunicación con el exterior a través de los periféricos. Estos periféricos pueden ser: de entrada, como los teclados; de salida, como los tubos de rayos catódicos, y de entrada y salida, como los discos magnéticos.

Unidad Central De Proceso.

La CPU constituye el cerebro de una computadora digital, pues realiza todas las operaciones aritméticas y lógicas sobre los datos y además controla todos los procesos que se desarrollan en la computadora. Por ejemplo, para que se ejecute un a instrucción, ésta debe estar en el interior de la CPU, concretamente en la UC y si hay que realizar cálculos, interviene la UAL. Veamos como funciona cada una de ellas.

Unidad de Control.

Para realizar su tarea la UC necesita conocer, por un lado, la instrucción y, por otro, una serie de informaciones adicionales que deberá tener en cuenta para coordinar, de forma correcta, la ejecución de la instrucción. El resultado de la interpretación de dichas informaciones son una serie de órdenes a los diferentes elementos de la computadora.

La UC no emite todas las órdenes a la vez, sino siguiendo una determinada secuencia. Para ello utiliza un elemento que le va indicando el instante en que debe ejecutar una determinada fase de la instrucción. A este elemento se le denomina Reloj, y se dice que sincroniza las acciones de la UC; cuanto más rápido marque el tiempo, más rápida será la ejecución de la instrucción. Sin embargo, hay un limite, ya que, si marca excesivamente rápido, es posible que no puedan cumplir adecuadamente las órdenes de los diferentes elementos, por lo que se producirán errores.

En la figura 2.2 se esquematiza el conjunto de señales que utiliza la UC y las que genera. Como informaciones adicionales a las instrucciones podemos ver los impulsos de reloj y los indicadores de estado. Los indicadores de estado son una serie de bits que se modifican según resultados de las operaciones anteriores guardando una memoria histórica de los acontecimientos precedentes para que, en función de dichos acontecimientos, pueda la UC tomar decisiones.

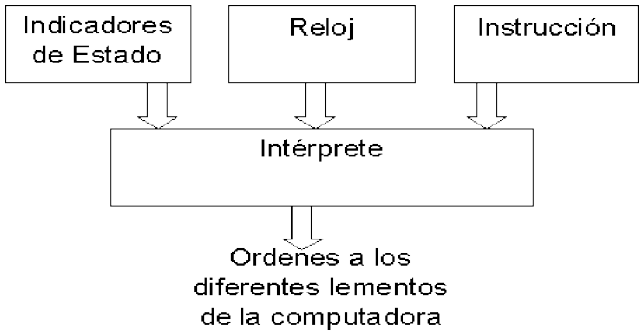


Figura 2.2. Señales que intervienen en la UC.

La unidad de control esta formada, básicamente por un elemento que interpreta las instrucciones y varios elementos de memoria denominados registros. Uno de estos registros almacena la instrucción mientras el intérprete está traduciendo su significado, por lo que se denomina Registro de Instrucción (RI). El resto de las instrucciones permanecen en la memoria, esperando que les toque su turno de ejecución.

La UC por otra parte deberá conocer cuál es la dirección de la próxima instrucción, para poder ir a buscarla una vez que finaliza la ejecución de la instrucción en curso; dirección que guarda el registro llamado Contador de Programa (CP).

Los indicadores de estado están agrupados en un registro denominado Registro de Estado (RE).

La Figura 2.3 muestra los elementos que acabamos de nombrar.

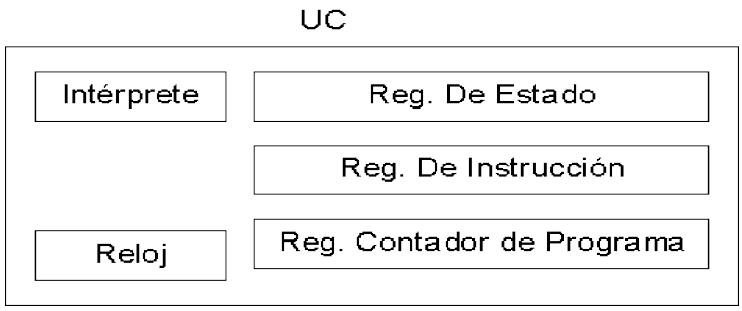


Figura 2.3. Elementos básicos de la Unidad de Control.

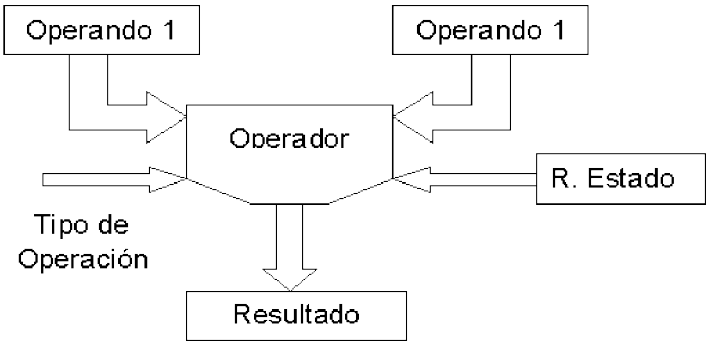
Unidad Aritmético - Lógica.

La unidad Aritmético - Lógica (UAL) es la encargada de realizar los cálculos. Los datos sobre los que se realizan la operaciones se denominan operandos. Al elemento encargado de ejecutar las operaciones se le denomina operador, y esta formado por una serie de circuitos electrónicos que son capaces de sumar dos números binarios o hacer las operaciones lógicas elementales: disyunción, conjunción y negación; incluso algunos operadores son también capaces de multiplicar, dividir y realizar otras operaciones mas complejas.

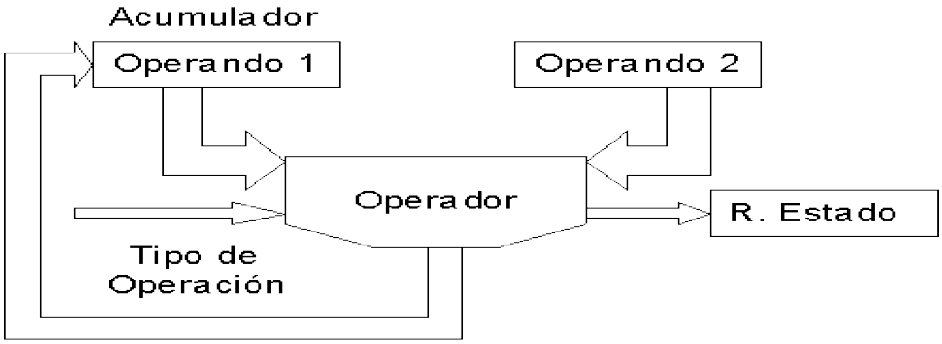
Para que el operador realice la operación, los operandos se llevan a la UAL y se guardan en unos registros denominados registros de trabajo. El resultado de la operación se guarda también en un registro antes de ser llevado a la memoria o a la Unidad de Entradas y Salidas. Frecuentemente se utiliza un mismo registro para guardar uno de los operandos y, también, el resultado, denominado registro Acumulador.

El operador, además de calcular el valor de la operación, modifica el registro de estado según el resultado de la operación. Así, si el resultado es un valor negativo, se modifica un bit de dicho registro, llamado bit negativo o bit N, poniéndose a 1; por el contrario, el bit N permanecerá en estado 0 mientras el contenido del acumulador no sea negativo. De igual forma indicara la UAL a la UC si el resultado ha sido cero, o si ha producido algún acarreo, etc.

En la figura 2.4 se muestran los elementos de la UAL y las señales que intervienen.



(a) UAL con tres registros: 2 para los operandos y 1 para el resultado.



(b) UAL con acumulador.

Figura 2.4. Elementos y señales de la AUL.

Unidad De Memoria Principal

La memoria principal esta formada por un conjunto de unidades llamadas palabras. Dentro de cada una de estas palabras se guarda la información que constituye una instrucción o parte de ella (puede darse el caso de que una sola instrucción necesite varia palabras), o un dato o parte de un dato (también un dato puede ocupar varias palabras).

A la cantidad de palabras que forman la MP se le denomina capacidad de memoria. De este modo, cuanto mayor sea el numero de palabras mayor será el numero de instrucciones y datos que podrá almacenar la computadora.

Una palabra esta formada a su vez de unidades mas elementales llamadas bits, del mismo modo que en el lenguaje natural una palabra esta formada por letras. Cada bit solo puede guardar dos valores, el valor 0 o el valor 1; por eso se dice que son elementos binarios.

El numero de bits que forman una palabra se llama longitud de palabra. Por regla general, las computadoras potentes tienen memorias con longitud de palabra grande, mientras que las computadoras pequeñas tienen memorias con longitud de palabra menor.

En la figura 2.5 se muestra como se puede estar organizada una Memoria Principal.

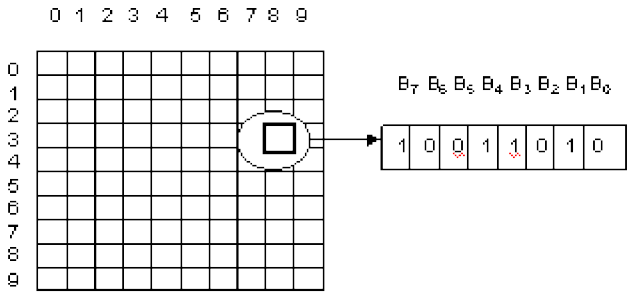


Figura 2.5. Organización de una unidad de memoria.

Las palabras forman una matriz de 10 filas y 10 columnas. La primera palabra corresponderá con la dirección 00, la segunda con la 01, y la ultima, con la 99. La capacidad de la memoria será de 10 * 10 = 100 palabras. También se muestra la longitud de la palabra 38, que es de 8 bits, al igual que las demás, y la información que contiene, que es el valor binario 10011010.

Las palabras se distinguen entre si por la posición que ocupan en la MP, y se puede guardar una información y luego recuperarla indicando el numero de dicha posición. A los números que señalan las posiciones de memoria se les da el nombre de direcciones de memoria.

La acción de guardar una información en una palabra de la memoria se llama acceso de escritura, y la acción de recuperarla, acceso de lectura. Los accesos son coordinados por la UC. La secuencia de ordenes que debe generar la UC se indica en la tabla 2.1.

En la tabla 2.1 (a) se muestra un acceso de escritura. Obsérvese que la UC debe indicar, además de la posición donde se debe guardar el dato, el valor del dato y las indicaciones de control que le digan a la memoria que se desea guardar el dato y el momento en que debe iniciarse la operación de escritura.

Esta ultima orden la dará la UC cuando este segura de que los datos anteriores han llegado correctamente a la MP. Después de esta ultima orden, la UC espera un tiempo para asegurar que se ha escrito la información en la MP.

En la tabla 2.1 (b) se muestra como se realiza un acceso de lectura. En este caso, la UC no indica el dato, puesto es precisamente lo que espera recibir. Los demás pasos son idénticos a los del acceso de escritura.

Desde que se inicia la secuencia hasta que finaliza transcurre un tiempo, denominado tiempo de acceso, cuya duración depende de la tecnología con que esta fabricada la MP.

Tabla 2.1. Secuencia de acceso a la memoria.

1	La UC indica a la MP que desea escribir.
2	La UC envía la dirección en la que se va a poner el dato.
3	La UC envía el valor del dato.
4	La UC da la orden de inicio del acceso.
5	La UC espera el tiempo necesario para que la MP realice la escritura.

(a) Acceso de escritura.

1	La UC indica a la MP que desea leer.
2	La UC envía a la dirección donde está el dato.
3	La UC da la orden de inicio del acceso.
4	La UC espera el tiempo necesario para que la MP realice la lectura.

(b) Acceso de lectura.

REGISTROS DEL PROCESADOR.

Los registros del procesador se emplean para controlar instrucciones en ejecución, manejar direccionamiento de memoria y proporcionar capacidad aritmética. Los registros son direccionables por medio de un nombre. Los bits por convención, se numeran de derecha a izquierda, como en:

... 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Registros de segmento

Un registro de segmento tiene 16 bits de longitud y facilita un área de memoria para direccionamiento conocida como el segmento actual.

Registro CS. El DOS almacena la dirección inicial del segmento de código de un programa en el registro CS. Esta dirección de segmento, mas un valor de desplazamiento en el registro apuntador de instrucción (IP), indica la dirección de una instrucción que es buscada para su ejecución.

Registro DS. La dirección inicial de un segmento de datos de programa es almacenada en el registro DS. En términos sencillos, esta dirección, mas un valor de desplazamiento en una instrucción, genera una referencia a la localidad de un byte especifico en el segmento de datos.

Registro SS. El registro SS permite la colocación en memoria de una pila, para almacenamiento temporal de direcciones y datos. El DOS almacena la dirección de inicio del segmento de pila de un programa en le registro SS. Esta dirección de segmento, mas un valor de desplazamiento en el registro del apuntador de pila (SP), indica la palabra actual en la pila que esta siendo direccionada.

Registros ES. Alguna operaciones con cadenas de caracteres (datos de caracteres) utilizan el registro extra de segmento para manejar el direccionamiento de memoria. En este contexto, el registro ES esta asociado con el registro DI (índice). Un programa que requiere el uso del registro ES puede inicializarlo con una dirección de segmento apropiada.

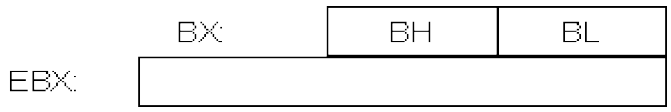
Registros FS y GS. Son registros extra de segmento en los procesadores 80386 y posteriores.

Registros de propósito general.

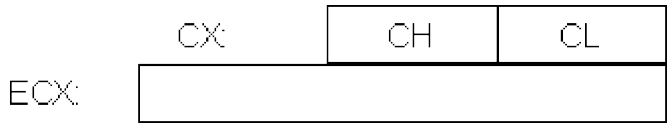
Los registros de propósito general AX, BX, CX y DX son los caballos de batalla del sistema. Son únicos en el sentido de que se puede direccionarlos como una palabra o como una parte de un byte. El ultimo byte de la izquierda es la parte "alta", y el ultimo byte de la derecha es la parte "baja". Por ejemplo, el registro CX consta de una parte CH (alta) y una parte Cl (baja), y usted puede referirse a cualquier parte por su nombre.

Registro AX. El registro AX, el acumulador principal, es utilizado para operaciones que implican entrada/salida y la mayor parte de la aritmética. Por ejemplo, las instrucciones para multiplicar , dividir y traducir suponen el uso del AX. También, algunas operaciones generan código mas eficiente si se refieren al AX en lugar de a los otros registros.

Registro BX. El BX es conocido como el registro base ya que es el único registro de propósito general que puede ser índice para direccionamiento indexado. También es común emplear el BX para cálculos.



Registro DX. El DX es conocido como l registro de datos. Alguna operaciones de entrada/salida requieren uso, y las operaciones de multiplicación y división con cifras grandes suponen al DX y al AX trabajando juntos.



Pude usar los registros de propósito general para suma y resta de cifras de 8, 16 o 32 bits.

Registro de Apuntador de Instrucciones.

El registro apuntador de instrucciones (IP) de 16 bits contiene el desplazamiento de dirección de la siguiente instrucción que se ejecuta. El IP esta asociado con el registro CS en el sentido de que el IP indica la instrucción actual dentro del segmento de código que se esta ejecutando actualmente. Los procesadores 80386 y posteriores tienen un IP ampliado de 32 bits, llamado EIP.

En el ejemplo siguiente, el registro CS contiene 25A4[0]H y el IP contiene 412H. Para encontrar la siguiente instrucción que será ejecutada, el procesador combina las direcciones en el CS y el IP:

Segmento de dirección en el registro CS: 25A40H
Desplazamiento de dirección en el registro IP: + 412H
Dirección de la siguiente instrucción: 25E52H

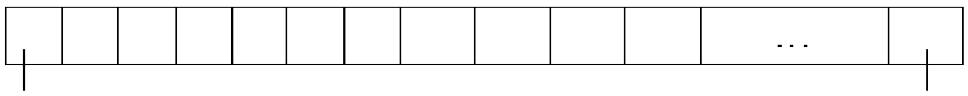
Registros Apuntadores.

Los registros SP (apuntador de la pila) Y BP (apuntador de base) están asociados con el registro SS y permiten al sistema accesar datos en el segmento de la pila.

Registro SP. El apuntador de la pila de 16 bits esta asociado con el registro SS y proporciona un valor de desplazamiento que se refiere a la palabra actual que esta siendo procesada en la pila. Los procesador es 80386 y posteriores tienen un apuntador de pila de 32 bits, el registro ESP. El sistema maneja de forma automática estos registros.

En el ejemplo siguiente, el registro SS contiene la dirección de segmento 27B3[0]H y el SP el desplazamiento 312H. Para encontrar la palabra actual que esta siendo procesada en la pila, la computadora combina las direcciones en el SS y el SP:

Dirección de semento en el registro SS: 27B30H
Desplazamiento en el registro SP: +312H
Dirección en la pila: 27E42H



27B3[0]H
Dirección del segmento SS

312H
Desplazamiento del SP

Registro BP. El BP de 16 bits facilita la referencia de parámetros, los cuales son datos y direcciones transmitidos vía pila. Los procesadores 80386 y posteriores tienen un BP ampliado de 32 bits llamado el registro EBP.

Registros Indice.

Los registros SI y DI están disponibles para direccionamiento indexado y para sumas y restas.

Registro SI. El registro índice fuente de 16 bits es requerido por algunas operaciones con cadenas (de caracteres). En este contexto, el SI esta asociado con el registro DS. Los procesadores 80386 y posteriores permiten el uso de un registro ampliado de 32 bits, el ESI.

Registro DI. El registro índice destino también es requerido por algunas operaciones con cadenas de caracteres. En este contexto, el DI esta asociado con el registro ES. Los procesadores 80386 y posteriores permiten el uso de un registro ampliado de 32 bits, el EDI.

Registro de Banderas.

De los 16 bits del registro de banderas, nueve son comunes a toda la familia de procesadores 8086, y sirven para indicar el estado actual de la maquina y el resultado del procesamiento. Muchas instrucciones que piden comparaciones y aritmética cambian el estado de las banderas, algunas cuyas instrucciones pueden realizar pruebas para determinar la acción subsecuente. En resumen, los bits de las banderas comunes son como sigue:

OF (Overflow, desbordamiento). Indica desbordamiento de un bit de orden alto (mas a la izquierda) después de una operación aritmética.

DF (dirección). Designa la dirección hacia la izquierda o hacia la derecha para mover o comparar cadenas de caracteres.

IF (interrupción). Indica que una interrupción externa, como la entrada desde el teclado, sea procesada o ignorada.

TF (trampa). Permite la operación del procesador en modo de un paso. Los

programas depuradores, como el DEBUG, activan esta bandera de manera que usted pueda avanzar en la ejecución de una sola instrucción a un tiempo, para examinar el efecto de esa instrucción sobre los registros de memoria.

SF (signo). Contiene el signo resultante de una operación aritmética (0 = positivo y 1 = negativo).

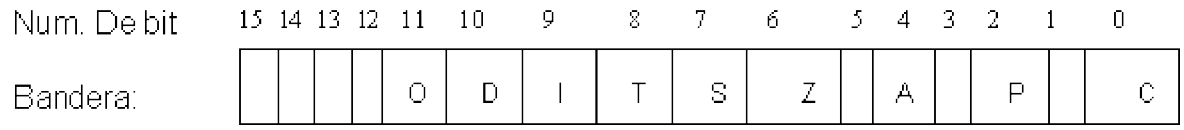
ZF (cero). Indica el resultado de una operación aritmética o de comparación (0 = resultado diferente de cero y 1 = resultado igual a cero).

AF (acarreo auxiliar). Contiene un acarreo externo del bit 3 en un dato de 8 bits para aritmética especializada.

PF (paridad). Indica paridad par o impar de una operación en datos de 8 bits de bajo orden (mas a la derecha).

CF (acarreo). Contiene el acarreo de orden mas alto (mas a la izquierda) después de una operación aritmética; también lleva el contenido del ultimo bit en una operación de corrimiento o de rotación.

Las banderas están en el registro de banderas en las siguientes posiciones:



Las banderas mas importantes para la programación en ensamblador son O, S, Z y C, para operaciones de comparación y aritméticas, y D para operaciones de cadenas de caracteres. Los procesadores 80286 y posteriores tienen algunas banderas usadas para propósitos internos, en especial las que afectan al modo protegido. Los procesadores 80286 y posteriores tienen un registro extendido de banderas conocido como Eflags.

UNIDAD DE ENTRADAS Y SALIDAS.

Ya se ha visto en las secciones precedentes como funcionan la CPU y la MP, pero puede decirse que es necesaria la comunicación entre el interior de la computadora y su entorno o periferia. Esta comunicación se consigue a través de dispositivos de muy diversos tipos, como son: teclados, impresoras, pantallas, discos magnéticos, etc. Es estos dispositivos se les conoce con el nombre genérico de periféricos.

En la figura 2.6 se muestran algunos periféricos conectados a la Unidad de E/S, la cual hace de intermediaria entre los periféricos y la CPU. Las flechas indican el sentido en que fluye la información.

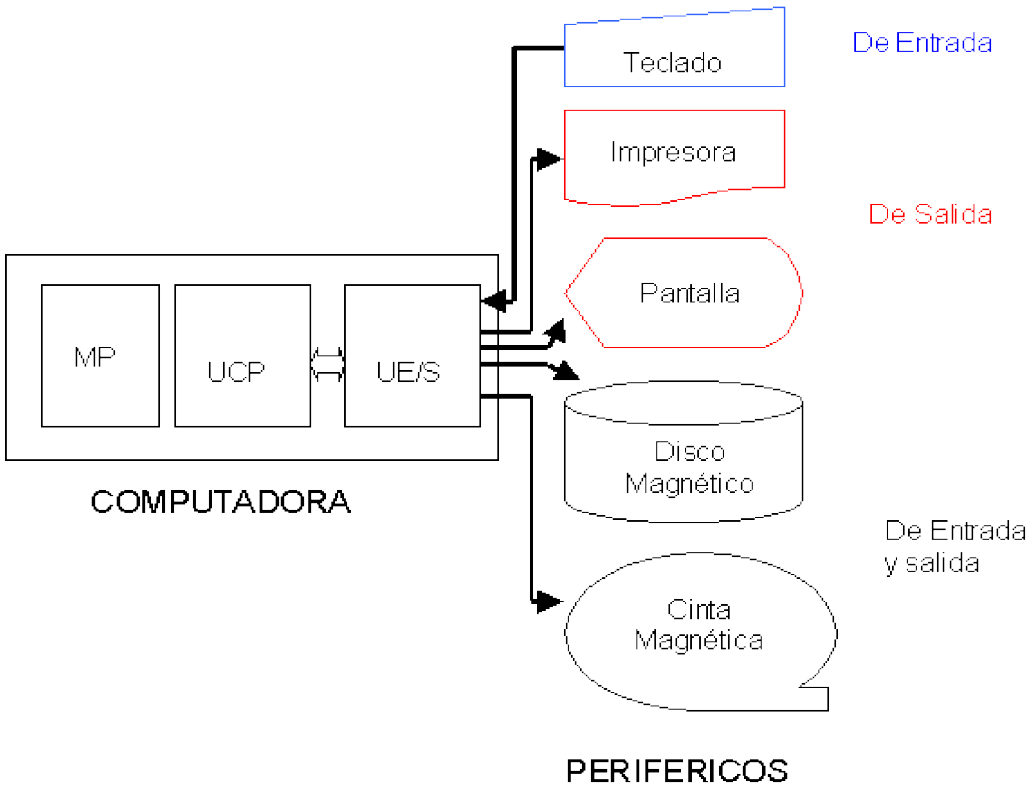


Figura 2.6. La unidad de E/S hace de intermediaria entre la UCP y los periféricos.

La coordinación de la comunicación entre los periféricos y la CPU la realiza la Unidad de E/S. Obsérvese que esta no es un periférico sino un dispositivo que gestiona a los periféricos siguiendo las ordenes de la CPU; es decir, la Unidad de E/S recibe de la Unidad de Control información sobre el tipo de transferencia de datos que debe realizar (si es de entrada o de salida) y periférico que debe de utilizar; si es de salida recibirá también el dato que debe enviar y el momento de la operación.

Entonces, la Unidad de E/S seleccionara el periférico y ejecutara la operación teniendo en cuanta las características propias de cada periférico. Una vez ejecutada la orden avisara a la UC de la terminación de la transferencia.

Cada periférico o parte de un periférico tendrá asignado un numero o dirección que servirá para identificarlo. Cuando la UC quiera seleccionarlo enviara dicho numero a la Unidad de E/S.

Para solucionar el problema de la imposibilidad de saber cuanto tiempo durara una transferencia de información con un periférico se han desarrollado diversas técnicas de comunicación entre la CPU y los periféricos.

Lo mas sencillo es que la CPU, cuando desee hacer una transferencia de información con un periférico, pregunte a la Unidad de E/S si dicho periférico se encuentra disponible. Si no lo esta, debe repetir la pregunta una y otra vez hasta obtener una respuesta afirmativa, en cuyo caso se inicia la transferencia de información.

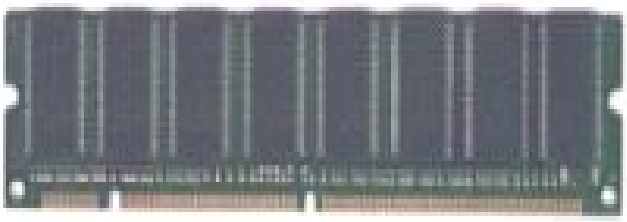
Si se desea obtener mayor rendimiento del ordenador, se puede emplear otro método que se denomina sincronización mediante interrupción. La característica de este método es que la CPU, en lugar de dedicarse a preguntar a la Unidad de E/S por el periférico que desea utilizar, lo que hace es indicar a la Unidad de E/S que desea hacer una transferencia con el periférico, y seguidamente, si no esta el periférico preparado, empieza otra tarea, olvidándose momentáneamente del periférico.

Cuando este preparado, la Unidad de E/S indicara a la CPU que puede realizarse la transferencia; entonces, la CPU interrumpirá la tarea que esta realizando y atenderá al periférico. De esta forma, la CPU no pierde tiempo esperando al periférico.

Qué es... la memoria RAM?

La memoria principal o RAM (Random Access Memory, Memoria de Acceso Aleatorio) es donde el computador guarda los datos que está utilizando en el momento presente. El almacenamiento es considerado temporal por que los datos y programas permanecen en ella mientras que la computadora este encendida o no sea reiniciada.

Se le llama RAM por que es posible acceder a cualquier ubicación de ella aleatoria y rápidamente Físicamente, están constituidas por un conjunto de chips o módulos de chips normalmente conectados a la tarjeta madre. Los chips de memoria son rectángulos negros que suelen ir soldados en grupos a unas plaquitas con "pines" o contactos:



La diferencia entre la RAM y otros tipos de memoria de almacenamiento, como los disquetes o los discos duros, es que la RAM es mucho más rápida, y que se borra al apagar el computador, no como los Disquetes o discos duros en donde la información permanece grabada.

Tipos de RAM

Hay muchos tipos de memorias DRAM, Fast Page, EDO, SDRAM, etc. Y lo que es peor, varios nombres. Trataremos estos cuatro, que son los principales, aunque mas adelante en este Informe encontrará prácticamente todos los demás tipos.
DRAM: Dinamic-RAM, o RAM DINAMICA, ya que es "la original", y por tanto la más lenta.

Usada hasta la época del 386, su velocidad típica es de 80 ó 70 nanosegundos (ns), tiempo éste que tarda en vaciarse para poder dar entrada a la siguiente serie de datos. Por ello, es más rápida la de 70 ns que la de 80 ns.
Físicamente, aparece en forma de DIMMs o de SIMMs, siendo estos últimos de 30 contactos.

Fast Page (FPM): a veces llamada DRAM (o sólo "RAM"), puesto que evoluciona directamente de ella, y se usa desde hace tanto que pocas veces se las diferencia. Algo más rápida, tanto por su estructura (el modo de Página Rápida) como por ser de 70 ó 60 ns.

Usada hasta con los primeros Pentium, físicamente aparece como SIMMs de 30 ó 72 contactos (los de 72 en los Pentium y algunos 486).

EDO: o EDO-RAM, Extended Data Output-RAM. Evoluciona de la Fast Page; permite empezar a introducir nuevos datos mientras los anteriores están saliendo (haciendo su Output), lo que la hace algo más rápida (un 5%, más o menos).
Muy común en los Pentium MMX y AMD K6, con velocidad de 70, 60 ó 50 ns. Se instala sobre todo en SIMMs de 72 contactos, aunque existe en forma de DIMMs de 168.

SDRAM: Sincronic-RAM. Funciona de manera sincronizada con la velocidad de la placa (de 50 a 66 MHz), para lo que debe ser rapidísima, de unos 25 a 10 ns. Sólo se presenta en forma de DIMMs de 168 contactos; es usada en los Pentium II de menos de 350 MHz y en los Celeron.

PC100: o SDRAM de 100 MHz. Memoria SDRAM capaz de funcionar a esos 100 MHz, que utilizan los AMD K6-2, Pentium II a 350 MHz y computadores más modernos; teóricamente se trata de unas especificaciones mínimas que se deben cumplir para funcionar correctamente a dicha velocidad, aunque no todas las memorias vendidas como "de 100 MHz" las cumplen.

PC133: o SDRAM de 133 MHz. La más moderna (y recomendable).

SIMMs y DIMMs
Se trata de la forma en que se juntan los chips de memoria, del tipo que sean, para conectarse a la placa base del ordenador. Son unas plaquitas alargadas con conectores en un extremo; al conjunto se le llama módulo.

El número de conectores depende del bus de datos del microprocesador, que más que un autobús es la carretera por la que van los datos; el número de carriles de dicha carretera representaría el número de bits de información que puede manejar cada vez.

SIMMs: Single In-line Memory Module, con 30 ó 72 contactos. Los de 30 contactos pueden manejar 8 bits cada vez, por lo que en un 386 ó 486, que tiene un bus de datos de 32 bits, necesitamos usarlos de 4 en 4 módulos iguales. Miden unos 8,5 cm (30 c.) ó 10,5 cm (72 c.) y sus zócalos suelen ser de color blanco.

Los SIMMs de 72 contactos, más modernos, manejan 32 bits, por lo que se usan de 1 en 1 en los 486; en los Pentium se haría de 2 en 2 módulos (iguales), porque el bus de datos de los Pentium es el doble de grande (64 bits).

DIMMs: más alargados (unos 13 cm), con 168 contactos y en zócalos generalmente negros; llevan dos muescas para facilitar su correcta colocación. Pueden manejar 64 bits de una vez, por lo que pueden usarse de 1 en 1 en los Pentium, K6 y superiores. Existen para voltaje estándar (5 voltios) o reducido (3.3 V).

Y podríamos añadir los módulos SIP, que eran parecidos a los SIMM pero con frágiles patitas soldadas y que no se usan desde hace bastantes años, o cuando toda o parte de la memoria viene soldada en la placa (caso de algunos ordenadores de marca).

Otros tipos de RAM

BEDO (Burst-EDO): una evolución de la EDO, que envía ciertos datos en "ráfagas". Poco extendida, compite en prestaciones con la SDRAM.
Memorias con paridad: consisten en añadir a cualquiera de los tipos anteriores un chip que realiza una operación con los datos cuando entran en el chip y otra cuando salen. Si el resultado ha variado, se ha producido un error y los datos ya no son fiables.

Dicho así, parece una ventaja; sin embargo, el ordenador sólo avisa de que el error se ha producido, no lo corrige. Es más, estos errores son tan improbables que la mayor parte de los chips no los sufren jamás aunque estén funcionando durante años; por ello, hace años que todas las memorias se fabrican sin paridad.
ECC: memoria con corrección de errores. Puede ser de cualquier tipo, aunque sobre todo EDO-ECC o SDRAM-ECC. Detecta errores de datos y los corrige; para aplicaciones realmente críticas. Usada en servidores y mainframes.

Memorias de Vídeo: para tarjetas gráficas. De menor a mayor rendimiento, pueden ser: DRAM -> FPM -> EDO -> VRAM -> WRAM -> SDRAM -> SGRAM

Memoria rom

ROM (memoria inalterable): Los ordenadores contienen casi siempre una cantidad pequeña de memoria de solo lectura que guarde las instrucciones para iniciar el ordenador. En la memoria ROM no se puede escribir.
PROM (memoria inalterable programable): Un PROM es un chip de memoria en la cual usted puede salvar un programa. Pero una vez que se haya utilizado el PROM, usted no puede reusarlo para salvar algo más. Como las ROM, los PROMS son permanentes.

EPROM (memoria inalterable programable borrable): Un EPROM es un tipo especial de PROM que puede ser borrado exponiéndolo a la luz ultravioleta.
EEPROM (eléctricamente memoria inalterable programable borrable): Un EEPROM es un tipo especial de PROM que puede ser borrado exponiéndolo a una carga eléctrica.

ROM, siglas para la memoria inalterable, memoria de computadora en la cual se han grabado de antemano los datos. Una vez que los datos se hayan escrito sobre un chip ROM, no pueden ser quitados y pueden ser leídos solamente.
Distinto de la memoria principal (RAM), la ROM conserva su contenido incluso cuando el ordenador se apaga. ROM se refiere como siendo permanente, mientras que la RAM es volátil.

La mayoría de los ordenadores personales contienen una cantidad pequeña de ROM que salve programas críticos tales como el programa que inicia el ordenador. Además, las ROM se utilizan extensivamente en calculadoras y dispositivos periféricos tales como impresoras láser, cuyas fuentes se salvan a menudo en las ROM.

Una variación de una ROM es un PROM (memoria inalterable programable). PROM son manufacturados como chips en blanco en los cuales los datos pueden ser escritos con dispositivo llamado programador de PROM.

CONCEPTO DE INTERRUPCION.

Una interrupción es una operación que suspende la ejecución de un programa de modo que el sistema pueda realizar una acción especial. La rutina de interrupción ejecuta y por lo regular regresa el control al procedimiento que fue interrumpido, el cual entonces reasume su ejecución.

TABLA DE SERVICIO DE INTERRUPCION

Cuando la computadora se enciende, el BIOS y el DOS establecen una tabla de servicios de interrupción en las localidades de memoria 000H-3FFH. La tabla permite el uso de 256 (100H) interrupciones, cada una con un desplazamiento:segmento relativo de cuatro bytes en la forma IP:CS.

El operando de una instrucción de interrupción como INT 05H identifica el tipo de solicitud. Como existen 256 entradas, cada una de cuatro bytes, la tabla ocupa los primeros 1, 024 bytes de memoria, desde 000H hasta 3FFH. Cada dirección en la tabla relaciona a una ruina de BIOS o del DOS para un tipo especifico de interrupción. Por lo tanto los bytes 0-3 contienen la dirección para la interrupción 0, los bytes 4-7 para la interrupción 1, y así sucesivamente:

INT 00H	INT 01H	INT 02H	INT 03H	INT 04H	INT 05H	INT 06H	. . .
IP:CS	IP:CS	IP:CS	IP:CS	IP:CS	IP:CS	IP:CS	. . .
00H	04H	08H	0CH	10H	14H	18H	. . .

TIPOS DE INTERRUPCIONES.

Las interrupciones se dividen en dos tipos las cuales son: Externas y Internas. Una interrupción externa es provocada por un dispositivo externo al procesador. Las dos líneas que pueden señalar interrupciones externas son la línea de interrupción no enmascarable (NMI) y la línea de petición de interrupción (INTR).

La línea NMI reporta la memoria y errores de paridad de E/S. El procesador siempre actúa sobre esta interrupción, aun si emite un CLI para limpiar la bandera de interrupción en un intento por deshabilitar las interrupciones externas. La línea INTR reporta las peticiones desde los dispositivos externos, en realidad, las interrupciones 05H a la 0FH, para cronometro, el teclado, los puertos seriales, el disco duro, las unidades de disco flexibles y los puertos paralelos.

Una interrupción interna ocurre como resultado de la ejecución de una instrucción INT o una operación de división que cause desbordamiento, ejecución en modo de un paso o una petición para una interrupción externa, tal como E/S de disco. Los programas por lo común utilizan interrupciones internas, que no son enmascarables, para accesar los procedimientos del BIOS y del DOS.

INTERRUPCION DE BIOS.

El BIOS contiene un extenso conjunto de rutinas de entrada/salida y tablas que indican el estado de los dispositivos del sistema. El dos y los programas usuarios pueden solicitar rutinas del BIOS para la comunicación con los dispositivos conectados al sistema. El método para realizar la interfaz con el BIOS es el de las interrupciones de software. A continuación se listan algunas interrupciones del BIOS.

INT 00H: División entre cero. Llamada por un intento de dividir entre cero. Muestra un mensaje y por lo regular se cae el sistema.
INT 01H: Un solo paso. Usado por DEBUG y otros depuradores para permitir avanzar por paso a través de la ejecución de un programa.

INT 02H: Interrupción no enmascarare. Usada para condiciones graves de hardware, tal como errores de paridad, que siempre están habilitados. Por lo tanto un programa que emite una instrucción CLI (limpiar interrupciones) no afecta estas condiciones.

INT 03H: Punto de ruptura. Usado por depuración de programas para detener la ejecución.

INT 04H: Desbordamiento. Puede ser causado por una operación aritmética, aunque por lo regular no realiza acción alguna.

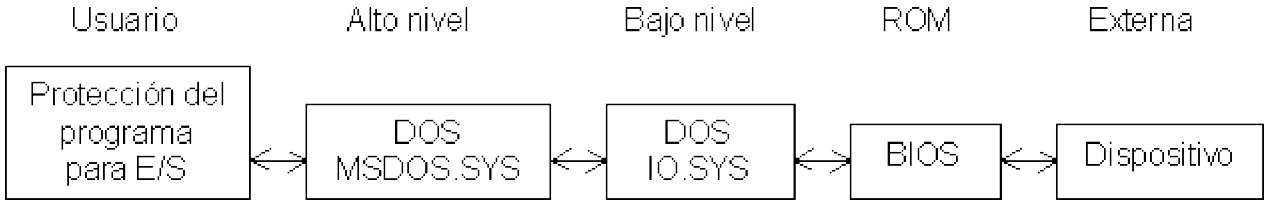
INT 05H: Imprime pantalla. Hace que el contenido de la pantalla se imprima. Emita la INT 05H para activar la interrupción internamente, y presione las teclas Cltr + PrtSC para activarla externamente. La operación permite interrupciones y guarda la posición del cursor.

INT 08H: Sistema del cronometro. Una interrupción de hardware que actualiza la hora del sistema y (si es necesario) la fecha. Un chip temporizador programable genera una interrupción cada 54.9254 milisegundos, casi 18.2 veces por segundo.

INT 09H: Interrupción del teclado. Provocada por presionar o soltar una tecla en el teclado.
INT OBH, INT OCH: Control de dispositivo serial. Controla los puertos COM1 y COM2, respectivamente.
INT 0DH, INT OFH: Control de dispositivo paralelo. Controla los puertos LPT1 y LPT2, respectivamente.
INT 0EH: Control de disco flexible. Señala actividad de disco flexible, como la terminación de una operación de E/S.
INT 10H: Despliegue en vídeo. Acepta el numero de funciones en el AH para el modo de pantalla, colocación del cursor, recorrido y despliegue.
INT 11H: Determinación del equipo. Determina los dispositivos opcionales en el sistema y regresa el valor en la localidad 40:10H del BIOS al AX. (A la hora de encender el equipo, el sistema ejecuta esta operación y almacena el AX en la localidad 40:10H).
INT 12H: Determinación del tamaño de la memoria. En el AX, regresa el tamaño de la memoria de la tarjeta del sistema, en términos de kilobytes contiguos.
INT 13H: Entrada/salida de disco. Acepta varias funciones en el AH para el estado del disco, sectores leídos, sectores escritos, verificación, formato y obtener diagnostico.

INTERRUPCION DEL DOS.

Los dos módulos del DOS, IO.SYS y MSDOS.SYS, facilitan el uso del BIOS. Ya que proporcionan muchas de las pruebas adicionales necesarias, las operaciones del DOS por lo general son mas fáciles de usar que sus contrapartes del BIOS y por lo común son independientes de la maquina.
IO.SYS es una interfaz de nivel bajo con el BIOS que facilita la lectura de datos desde la memoria hacia dispositivos externos.
MSDOS.SYS contiene un administrador de archivos y proporciona varios servicios. Por ejemplo, cuando un programa usuario solicita la INT 21H, la operación envía información al MSDOS.SYS por medio del contenido de los registros. Para completar la petición, MSDOS.SYS puede traducir la información a una o mas llamadas a IO.SYS, el cual a su vez llama al BIOS. Las siguientes son las relaciones implícitas:



INTERUPCIONES DEL DOS.

Las interrupciones desde la 20H hasta la 3FH están reservadas para operaciones del DOS. A continuación se mencionan algunas de ellas.

INT 20H: Termina programa. Finaliza la ejecución de un programa .COM, restaura las direcciones para Cltr + Break y errores críticos, limpia los bufer de registros y regresa el control al DOS. Esta función por lo regular seria colocada en el procedimiento principal y al salir de el, CS contendría la dirección del PSP. La terminación preferida es por medio de la función 4CH de la INT 21H.

INT 21H: Petición de función al DOS. La principal operación del DOS necesita una función en el AH.

INT 22H: Dirección de terminación. Copia la dirección de esta interrupción en el PSP del programa (en el desplazamiento 0AH) cuando el DOS carga un programa para ejecución. A la terminación del programa, el DOS transfiere el control a la dirección de la interrupción. Sus programas no deben de emitir esta interrupción.

INT 23H: Dirección de Cltr + Break. Diseñada para transferir el control a una rutina del DOS (por medio del PSP desplazamiento 0EH) cuando usted presiona Ctlt + Break o Ctlr + c. La rutina finaliza la ejecución de un programa o de un archivo de procesamiento por lotes. Sus programas no deben de emitir esta interrupción.

INT 24H: Manejador de error critico. Usada por el dos para transferir el control (por medio del PSP desplazamiento 12H) cuando reconoce un error critico (a veces una operación de disco o de la impresora).Sus programas no deben de emitir esta interrupción.

INT 25H: Lectura absoluta de disco. Lee el contenido de uno o mas sectores de disco.

INT 26H: Escritura absoluta de disco. Escribe información desde la memoria a uno o mas sectores de disco.

INT 27H: Termina pero permanece residente (reside en memoria). Hace que un programa .COM al salir permanezca residente en memoria.

INT 2FH: Interrupción de multiplexion. Implica la comunicación entre programas, como la comunicación del estado de un spooler de la impresora, la presencia de un controlador de dispositivo o un comando del DOS tal como ASSIGN o APPEND.

INT 33H: Manejador del ratón. Proporciona servicios para el manejo del ratón.

PROGRAMA DEBUG COMANDOS PRINCIPALES

* Q (Quit): permite abandonar el programa y volver al DOS.

* D [<dirección> [numbytes]] (dump): visualiza el contenido de la memoria. SYMDEB permite además visualizarla en palabras (DW), dobles palabras (DD), coma flotante ...

* A [<dirección>] (assemble): permite ensamblar a partir de CS:IP si no se indica una dirección concreta. Se admiten las directivas DB y DW del ensamblador. Las instrucciones que requieran indicar un registro de segmento, con DEBUG hay que ponerlas en una sola línea. Por ejemplo:

```
XLAT  CS:                          ; mal ensamblado con DEBUG (no así con SYMDEB)
MOV   WORD PTR ES:[100],1234  ; error en DEBUG (sí vale con SYMDEB)
CS:                                ; bien emsamblado con ambos
      XLAT
ES:                                ; y esto también
MOV   WORD PTR [100],1234
```

Los saltos inter-segmento deben especificarse como FAR (ej., CALL FAR [100]) a no ser que sea evidente que lo son (ej. CALL 1234:5678).

* E <dirección> [<lista>] (enter): permite consultar y modificar la memoria, byte a byte. Por ejemplo, con E 230 1,2,3 se introducirían los bytes 1, 2 y 3 a partir de DS:230. Si no se indica <lista>, se visualizará la memoria byte a byte, pudiéndose modificar los bytes deseados, avanzar al siguiente (barra espaciadora) o retroceder al anterior (signo -). Para acabar se pulsa RETURN.

* U [<direccion>[<rango>]] (unassemble): desensambla la memoria. Como ejemplos válidos: U ES:100, U E000:1940 ... si se indica rango, DEBUG desensamblará ese número de bytes y SYMDEB ese número de líneas. Por defecto se emplea CS: como registro de segmento.

* R [<registro>] (register): permite visualizar y modificar el valor de los registros. Por ejemplo, si se ejecuta la orden 'rip', se solicitará un nuevo valor para IP; con RF se muestran los flags y se permite modificar alguno:

Flag	Activo	Borrado
Desbordamiento	OV	NV
Dirección	DN (v)	UP (^)
Interrupción	EI	DI

Signo	NG (<0)	PL (>0)
Cero	ZR (=0)	NZ (!=0)
Acarreo auxiliar	AC	NA
Paridad	PE (par)	PO (impar)
Acarreo	CY	NC

* G [=<dirección> [,<dirección>,...]] (go): ejecuta código desde CS:IP (a menos que se indique una dirección concreta). Si se trabaja sobre memoria ROM no debe indicarse la segunda dirección. Para que el flujo del programa se detenga en la 2ª dirección o posteriores debe pasar necesariamente por ella(s). Se puede indicar hasta 10 direcciones donde debe detenerse.

* T [<veces>] (trace): ejecuta una instrucción del programa (a partir de CS:IP) mostrando a continuación el estado de los registros y la siguiente instrucción. Ejecutar T10 equivaldría a ejecutar 16 veces el comando T. Si la instrucción es CALL o INT, se ejecutará como tal introduciéndose en la subrutina o servidor de interrupciones correspondiente (SYMDEB no entra en los INT 21h).

* P [<veces>] (proceed): similar al comando T, pero al encontrarse un CALL o INT lo ejecuta de golpe sin entrar en su interior (ojo, ¡esto último falla al tracear sobre memoria ROM!).

* N <especificacion_fichero> (name): se asigna un nombre al programa que está siendo creado o modificado. Se puede indicar la trayectoria de directorios.

* L [<dirección>] (load): carga el fichero de nombre indicado con el comando N. Si es ejecutable lo prepara adecuadamente para su inmediata ejecución. En BX:CX queda depositado el tamaño del fichero (BX=0 para ficheros de menos de 64 Kb). Por defecto, la dirección es CS:100h.

* L <dirección> <unidad> <primer_sector><num_sectores> (load): carga sectores de la unidad 0, 1, ... (A, B, ...) a memoria. Se trata de sectores lógicos del DOS y no los sectores físicos de la BIOS. Las versiones antiguas de SYMDEB dan errores en particiones de más de 32 Mb.

* W [<dirección>] (write): graba el contenido de una zona de memoria a disco. Si no se indica la dirección, se graba desde CS:100h hasta CS:100h+número_bytes; el número de bytes se indica en BX:CX (no es una dirección segmentada sino un valor de 32 bits). Si se trata de un EXE no se permitirá grabarlo (para modificarlos, hay que renombrarles para cambiarles la extensión, aunque de esta manera no serán montados al cargarlos).

* W <dirección> <unidad> <primer_sector><num_sectores> (write): graba sectores de la memoria a disco en la unidad 0, 1, ... (A, B, ...). Se trata de sectores lógicos del DOS y no los sectores físicos de la BIOS. Las versiones antiguas de SYMDEB dan errores en particiones de disco duro de más de 32 Mb.

* S <rango> <lista> (search): busca una cadena de bytes por la memoria. Para buscar la cadena "PEPE" terminada por cero en un área de 512 bytes desde DS:100 se haría: S 100 L 200 "PEPE",0 (por defecto se busca en DS:). No se encontraría sin embargo "pepe" (en minúsculas).

* F <rango> <lista> (fill): llena la zona de memoria especificada con repeticiones de la lista de bytes indicada. Por ejemplo, para rellenar códigos 0AAh 100h bytes a partir de 9800h:0 se ejecutaría F 9800:0 L 100 AA; en vez de AA se podría haber indicado una lista de bytes o cadenas de caracteres.

* C <rango> <dirección> (compare): compara dos zonas de memoria mostrando las diferencias. Por ejemplo, para comparar 5 bytes de DS:100 y DS:200 se hace: C 100 L 5 200.

* M <rango> <dirección> (move): Más que mover, copia una zona de memoria en otra de manera inteligente (controlando los posibles solapamientos de los bloques).

* I <puerto> (input): visualiza la lectura del puerto de E/S indicado.

* O <puerto> <valor> (output): envia un valor a un puerto de E/S.

* H <valor1> <valor2> (hexaritmetic): muestra la suma y resta de valor1 y valor2, ambos operandos de un máximo de 16 bits (si hay desbordamiento se trunca el resultado, que tampoco excede los 16 bits).

Intruciones logicas

La lógica booleana es importante en el diseño de circuitos y tiene un paralelo en la lógica de programación. Las instrucciones para lógica booleana son AND, OR, XOR, TEST y NOT, que pueden usarse para poner bits en 0 o en 1 y para manejar datos ASCII con propósitos aritméticos. El formato general para las operaciones booleanas es:

[etiqueta :] | operación | {registro/memoria}, {registro/memoria/inmediato}|

El primer operando se refiere a un byte o palabra en un registro o memoria y es el único valor que es cambiado. El segundo operando hace referencia a un registro o a un valor inmediato. La operación compara los bits de los dos operandos referenciados y de acuerdo con esto establece las banderas CF, OF, PF, SF y ZF.

AND. Si ambos bits comparados son 1, establece el resultado en 1. Las demás condiciones dan como resultado 0.

OR. Si cualquiera (o ambos) de los bits comparados es 1, el resultado es 1. Si ambos bits están en 0, el resultado es 0.

XOR. Si uno de los bits comparados es 0 y el otro 1, el resultado es 1. Si ambos bits comparados son iguales (ambos 0 o ambos 1), el resultado es 0.

TEST. Establece las banderas igual que lo hace AND, pero no cambia los bits de los operandos.

Las operaciones siguientes AND, OR y XOR ilustran los mismos valores de bits como operandos:

	AND	OR	XOR
	0101	0101	0101
	0011	0011	0011
Resultado:	001	0111	0110

Es útil recordar la siguiente regla: el empleo de AND con bits 0 es 0 y el de OR con bits 1 es 1.

Ejemplos de operaciones booleanas.

Para los siguientes ejemplos independientes, suponga que AL contiene 11000101 y el BH contiene 01011100:

```
1.-  AND   AL,BH    ;Establece AL a 0100 0100
2.-  AND   AL,00H   ;Establece AL a 0000 0000
3.-  AND   AL,0FH   ;Establece AL a 0000 0101
4.-  OR    BH,AL    ;Establece BH a 1101 1101
5.-  OR    CL,CL    ;Pone en uno SF y ZF
6.-  XOR   AL,AL    ;Establece AL a 0000 0000
7.-  XOR   AL,0FFH  ;Establece AL a 0011 1010
```

Los ejemplos 2 y 6 muestran formas de limpiar un registro, y ponerlo a cero. El ejemplo 3 pone a cero los cuatro bits mas a la izquierda de AL.

TESt actúa igual que AND, pero solo establece las banderas. Aquí están algunos ejemplos :

```
1.- TEST  BL, 11110000B   ; Alguno de los bits de mas a la
    JNZ   ...             ; izquierda es BL en diferentes de cero?

2.- TEST  AL, 00000001B   ; AL contiene
    JNZ   ...             ; un numero impar?

3.- TEST  DX, 0FFH        ; El DX contiene
    JNZ   ...             ; un valor cero?
```

La instrucción NOT.

La instrucción NOT solo invierte los bits en un byte o palabra en un registro o en memoria; esto es, convierte los ceros en unos y los unos en ceros. El formato general es:

[etiqueta:]	NOT	{registro/memoria}

Por ejemplo si el AL contiene 11000101, la instrucción NOT AL cambia el AL a 00111010 (el resultado es el mismo de XOR AL, 0FFH). Las banderas no son afectadas.

Instrucciones aritméticas:

Instrucción	**Operandos**	**Función equivalente en "C"**	**Resultados**
ADD	op1, op2	op1 = op1 + op2	
SUB	op1, op2	op1 = op1 - op2	
ADC	op1, op2	op1 = op1 + op2 + Carry	
SBB	op1, op2	op1 = op1 - op2 - Carry	
INC	op1	op1 ++;	
DEC	op1	op1 --;	
MUL	op1	Si op1 es tipo byte:	AX = AL * op1
		Si op1 es tipo word:	(DX:AX) = AX * op1
DIV	op1	Si op1 es tipo byte:	AL = AX / op1
			AH = AX % op1
		Si op1 es tipo word:	AX = (DX : AX) / op1
			DX = (DX : AX) % op1
NEG	op1	op1 = - op1	

Manejo de bits (rotaciones y desplazamientos):

Instrucción	**Operandos**	**Función equivalente en "C"**
SHL	op1, cuenta	op1 <<= cuenta

SHR	op1, cuenta	op1 >>= cuenta
SAR	op1, cuenta	(no hay equivalente)
ROL	op1, cuenta	(no hay equivalente)
ROR	op1, cuenta	(no hay equivalente)

En todas las instrucciones de rotación/desplazamiento de bits, cuenta puede ser 1 o bien CX.

Operaciones lógicas (booleanas):

Instrucción	**Operandos**	**Función equivalente en "C"**
AND	op1, op2	op1 = op1 & op2
OR	op1, op2	op1 = op1 | op2
XOR	op1, op2	op1 = op1 ^ op2
NOT	op1	op1 = ~op1

```
EJEMPLO MACRO
:
INCLUDE MACRO.BIB
INCLUDE CARACTER.BIB
INCLUDE LIMPIAR.BIB
.MODEL SMALL
.STACK
.DATA

 TEXTO  DB 13,10,"MACRO Y BIBLIOTECA DE MACROS","$"
 TEXTO1 DB 13,10,"REGISTRO DEL MICROPROCESADOR", "$"
 TEXTO2 DB 13,10,"COMANDOS DEL DEBUG","$"
 TEXTO3 DB 13,10,"PRESIONE CUALQUIER TECLA PARA CONTINUAR","$"
 TEXTO4 DB 13,10,"INTERRUPCIONES DEL DOS Y BIOS","$"
 TEXTO5 DB 13,10,"LO QUE FALTA ES DE LA MATERIA LENGUAJE ENSAMBLADOR","$"
.CODE
 BEGIN PROC
 MOV AX,@DATA
 MOV DS,AX
 ;LIMPIA
  LETRA 83
  LETRA 32
  LETRA 89
  LETRA 32
  LETRA 66
  LETRA 73
  LETRA 79
  LETRA 83

  CADENA TEXTO
  CADENA TEXTO1
  CADENA TEXTO2
  CADENA TEXTO3
  CADENA TEXTO4
  CADENA TEXTO5

 mov AX,4C00H
 int 21h
 ENDP
END
```

UNIDAD 2: TRADUCTORES DE ALTO NIVEL

TRADUCTORES DE ALTO NIVEL

----DEFINICION Y EJEMPLOS DE TRADUCTORES DE ALTO NIVEL

Un traductor es un programa que toma el texto escrito en un lenguaje (el lenguaje fuente) y lo convierte en el texto equivalente en un segundo lenguaje (el lenguaje destino u objeto). Existen distintos tipos de traductores, entre ellos destacan:

- Ensambladores
- Preprocesadores
- Intérpretes
- Compiladores

----------COMPILADORES

Un **compilador** traduce desde un lenguaje de alto nivel a otro lenguaje de bajo nivel. Genera varias instrucciones de la máquina por cada comando fuente.

----------FACES DE UN COMPILADOR

Análisis Léxico.

En la fase de análisis léxico se leen los caracteres del programa fuente y se agrupan en cadenas que representan los componentes léxicos. Cada componente léxico es una secuencia lógicamente coherente de caracteres relativa a un identificador, una palabra reservada, un operador o un carácter de puntuación.

A la secuencia de caracteres que representa un componente léxico se le llama **lexema** (o con su nombre en inglés **token**). En el caso de los identificadores creados por el programador no solo se genera un componente léxico, sino que se genera otro lexema en la tabla de símbolos.

Análisis Sintáctico.

En esta fase, los componentes léxicos se agrupan en frases gramaticales que el compilador utiliza para sintetizar la salida.

Análisis Semántico.

La fase de análisis semántico se intenta detectar instrucciones que tengan la estructura sintáctica correcta, pero que no tengan significado para la operación implicada.

Generación de código Intermedio.

Algunos compiladores generan una representación intermedia explícita del programa fuente, una vez que se han realizado las fases de análisis. Se puede considerar esta operación intermedia como un subprograma para una máquina abstracta. Esta representación intermedia debe tener dos propiedades importantes: debe ser fácil de producir y fácil de traducir al programa objeto.

Optimización de Código.

En esta fase se trata de mejorar el código intermedio, de modo que resulte un código de máquina más rápido de ejecutar.

Generación de Código.

Esta constituye la fase final de un compilador. En ella se genera el código objeto que por lo general consiste en código en lenguaje máquina (código relocalizable) o código en lenguaje ensamblador.

Administrador de la tabla de símbolos.

Una tabla de símbolos es una estructura de datos que contiene un registro por cada identificador. El registro incluye los campos para los atributos del identificador.

El administrador de la tabla de símbolos se encarga de manejar los accesos a la tabla de símbolos, en cada una de las etapas de compilación de un programa.

Manejador de errores.

En cada fase del proceso de compilación es posibles encontrar errores. Es conveniente que el tratamiento de los errores se haga de manera centralizada a través de un manejador de errores. De esta forma podrán controlarse más eficientemente los errores encontrados en cada una de las fases de la compilación de un programa.

----INTERPRETE

Un **interprete** es un programa que acepta otro programa (el *programa fuente*) escrito en un determinado lenguaje (el *lenguaje fuente*), y lo ejecuta. El intérprete carga, analiza y ejecuta **una a una** las instrucciones del programa fuente.

---------DIFERENCIA ENTRE COMPILADOR Y INTERPRETE

Que el Interprete es un Programa que realiza varias fases del compilador, se diferencia de este, en que no genera código o no lo traduce a otro lenguaje, sino es ejecutado. Ademas La interpretación es muy lenta (hasta 100 veces mas)

----------APLICACIONES CON TRADUCTORES DE ALTO NIVEL

C++, PASCAL, JAVA, SQL, CLIPER, VISUAL BASIC, MYSQL, SQL SERVER

----------EJEMPLOS UTILIZANDO TRADUCTORES Y COMPILADORES

TRADUCTORES= los navegadores de internet: interpretan lenguaje html y lo muestran; Las líneas de comandos SQL: interpretan y ejecutan las consultas.
Algunos intérpretes conocidos son:
Un intérprete Caml
Un intérprete Lisp
El intérprete de comandos de Unix (*shell*).

COMPILADORES= década de los 50's, cuando con el advenimiento de computadoras comerciales surge también la necesidad de programarlas.

El primer compilador de FORTRAN tardó 18 años-persona en realizarse y era muy sencillo.

UNIDAD 3: SISTEMAS OPERATIVOS

DEFINICIÓN DE LOS SISTEMAS OPERATIVOS

Un **sistema operativo** (SO) es un conjunto de programas destinados a permitir la comunicación del usuario con un ordenador y gestionar sus recursos de manera eficiente. Comienza a trabajar cuando se enciende el ordenador, y gestiona el hardwarw de la máquina desde los niveles más básicos.

Un sistema operativo se puede encontrar normalmente en la mayoría de los aparatos electrónicos que podamos utilizar sin necesidad de estar conectados a un ordenador y que utilicen microprocesadores para funcionar, ya que gracias a estos podemos entender la máquina y que ésta cumpla con sus funciones (teléfonos móviles, reproductores de DVD, autoradios... y computadoras)

GENERACIONES DE LOS SISTEMAS OPERATIVOS

Los Sistemas Operativo al igual que el Hardware de los computadores, han sufrido una serie de cambios revolucionarios llamados generaciones En el caso del Hardware, las generaciones han sido marcadas por grandes avances en los componentes utilizados, pasando de válvulas (primera generación) a transistores (segunda generación), a circuitos integrados (tercera generación), a circuitos integrados de gran y muy gran escala (cuarta generación). Cada generación Sucesiva de hardware ha ido acompañada de reducciones substanciales en los costos, tamaño, emisión de calor y consumo de energía, y por incrementos notables en velocidad y capacidad.

Generacion Cero (década de 1940)

Los primeros sistemas computacionales no poseían sistemas operativos. Los usuarios tenían completo acceso al lenguaje de la maquina. Todas las instrucciones eran codificadas a mano.

Primera Generacion (década de 1950)

Los sistemas operativos de los años cincuenta fueron diseñados para hacer mas fluida la transición entre trabajos. Antes de que los sistemas fueran diseñados, se perdía un tiempo considerable entre la terminación de un trabajo y el inicio del siguiente. Este fue el comienzo de los sistemas de procesamiento por lotes, donde los trabajos se reunían por grupos o lotes. Cuando el trabajo estaba en ejecución, este tenia control total de la maquina. Al terminar cada trabajo, el control era devuelto al sistema operativo, el cual limpiaba y leía e iniciaba el trabajo siguiente.

Al inicio de los 50's esto había mejorado un poco con la introducción de tarjetas perforadas (las cuales servían para introducir los programas de lenguajes de máquina), puesto que ya no había necesidad de utilizar los tableros enchufables.

Además el laboratorio de investigación General Motors implementó el primer sistema operativo para la IBM 701. Los sistemas de los 50's generalmente ejecutaban una sola tarea, y la transición entre tareas se suavizaba para lograr la máxima utilización del sistema. Esto se conoce como sistemas de procesamiento por lotes de un sólo flujo, ya que los programas y los datos eran sometidos en grupos o lotes.

La introducción del transistor a mediados de los 50's cambió la imagen radicalmente.

Se crearon máquinas suficientemente confiables las cuales se instalaban en lugares especialmente acondicionados, aunque sólo las grandes universidades y las grandes corporaciones o bien las oficinas del gobierno se podían dar el lujo de tenerlas.

Para poder correr un trabajo (programa), tenían que escribirlo en papel (en Fortran o en lenguaje ensamblador) y después se perforaría en tarjetas. Enseguida se llevaría la pila de tarjetas al cuarto de introducción al sistema y la entregaría a uno de los operadores. Cuando la computadora terminara el trabajo, un operador se dirigiría a la impresora y desprendería la salida y la llevaría al cuarto de salida, para que la recogiera el programador.

Segunda Generación (a mitad de la década de 1960)

La característica de los sistemas operativos fue el desarrollo de los sistemas compartidos con multiprogramación, y los principios del multiprocesamiento. En los sistemas de multiprogramación, varios programas de usuario se encuentran al mismo tiempo en el almacenamiento principal, y el procesador se cambia rápidamente de un trabajo a otro. En los sistemas de multiprocesamiento se utilizan varios procesadores en un solo sistema computacional, con la finalidad de incrementar el poder de procesamiento de la maquina.

La independencia de dispositivos aparece después. Un usuario que desea escribir datos en una cinta en sistemas de la primera generación tenia que hacer referencia especifica a una unidad de cinta particular. En la segunda generación, el programa del usuario especificaba tan solo que un archivo iba a ser escrito en una unidad de cinta con cierto numero de pistas y cierta densidad.

Se desarrollo sistemas compartidos, en la que los usuarios podían acoplarse directamente con el computador a través de terminales. Surgieron sistemas de tiempo real, en que los computadores fueron utilizados en el control de procesos industriales. Los sistemas de tiempo real se caracterizan por proveer una respuesta inmediata.

Tercera Generación (mitad de década 1960 a mitad década de 1970)

Se inicia en 1964, con la introducción de la familia de computadores Sistema/360 de IBM. Los computadores de esta generación fueron diseñados como sistemas para usos generales . Casi siempre eran sistemas grandes, voluminosos, con el propósito de serlo todo para toda la gente. Eran sistemas de modos múltiples, algunos de ellos soportaban simultáneamente procesos por lotes, tiempo compartido, procesamiento de tiempo real y multiprocesamiento. Eran grandes y costosos, nunca antes se había construido algo similar, y muchos de los esfuerzos de desarrollo terminaron muy por arriba del presupuesto y mucho después de lo que el planificador marcaba como fecha de terminación.

Estos sistemas introdujeron mayor complejidad a los ambientes computacionales; una complejidad a la cual, en un principio, no estaban acostumbrados los usuarios.

Cuarta Generación (mitad de década de 1970 en adelante)

Los sistemas de la cuarta generación constituyen el estado actual de la tecnología. Muchos diseñadores y usuarios se sienten aun incómodos, después de sus experiencias con los sistemas operativos de la tercera generación.

Con la ampliación del uso de redes de computadores y del procesamiento en línea los usuarios obtienen acceso a computadores alejados geográficamente a través de varios tipos de terminales.

Los sistemas de seguridad se ha incrementado mucho ahora que la información pasa a través de varios tipos vulnerables de líneas de comunicación. La clave de cifrado esta recibiendo mucha atención; han sido necesario codificar los datos personales o de gran intimidad para que; aun si los datos son expuestos, no sean de utilidad a nadie mas que a los receptores adecuados.

El porcentaje de la población que tiene acceso a un computador en la década de los ochenta es mucho mayor que nunca y aumenta rápidamente.

El concepto de maquinas virtuales es utilizado. El usuario ya no se encuentra interesado en los detalles físicos de; sistema de computación que esta siendo accedida. En su lugar, el usuario ve un panorama llamado maquina virtual creado por el sistema operativo.

Los sistemas de bases de datos han adquirido gran importancia. Nuestro mundo es una sociedad orientada hacia la información, y el trabajo de las bases de datos es hacer que esta información sea conveniente accesible de una manera controlada para aquellos que tienen derechos de acceso.

DEFINICION DEL SISTEMA OPERATIVO

Un **sistema operativo** (SO) es un conjunto de programas destinados a permitir la comunicación del usuario con un ordenador y gestionar sus recursos de manera eficiente. Comienza a trabajar cuando se enciende el ordenador, y gestiona el hardware de la máquina desde los niveles más básicos.

Un sistema operativo se puede encontrar normalmente en la mayoría de los aparatos electrónicos que podamos utilizar sin necesidad de estar conectados a un ordenador y que utilicen microprocesadores para funcionar, ya que gracias a estos podemos entender la máquina y que ésta cumpla con sus funciones (teléfonos móviles, reproductores de DVD, autoradios... y computadoras).

FUNCIONES DE LOS SISTEMAS OPERATIVOS

1.- Aceptar todos los trabajos y conservarlos hasta su finalización.
2.- Interpretación de comandos: Interpreta los comandos que permiten al usuario comunicarse con el ordenador.
3.- Control de recursos: Coordina y manipula el hardware de la computadora, como la memoria, las impresoras, las unidades de disco, el teclado o el Mouse.
4.- Manejo de dispositivos de E/S: Organiza los archivos en diversos dispositivos de almacenamiento, como discos flexibles, discos duros, discos compactos o cintas magnéticas.
5.- Manejo de errores: Gestiona los errores de hardware y la pérdida de datos.
6.- Secuencia de tareas: El sistema operativo debe administrar la manera en que se reparten los procesos. Definir el orden. (Quien va primero y quien después).
7.- Protección: Evitar que las acciones de un usuario afecten el trabajo que esta realizando otro usuario.
8.- Multiacceso: Un usuario se puede conectar a otra máquina sin tener que estar cerca de ella.

9.- Contabilidad de recursos: establece el costo que se le cobra a un usuario por utilizar determinados recursos.

CARACTERÍSTICAS DE LOS SISTEMAS OPERATIVOS

En general, se puede decir que un Sistema Operativo tiene las siguientes características:

- Conveniencia. Un Sistema Operativo hace más conveniente el uso de una computadora.
- Eficiencia. Un Sistema Operativo permite que los recursos de la computadora se usen de la manera más eficiente posible.

- Habilidad para evolucionar. Un Sistema Operativo deberá construirse de manera que permita el desarrollo, prueba o introducción efectiva de nuevas funciones del sistema sin interferir con el servicio.
- Encargado de administrar el hardware. El Sistema Operativo se encarga de manejar de una mejor manera los recursos de la computadora en cuanto a hardware se refiere, esto es, asignar a cada proceso una parte del procesador para poder compartir los recursos.
- Relacionar dispositivos (gestionar a través del kernel). El Sistema Operativo se debe encargar de comunicar a los dispositivos periféricos, cuando el usuario así lo requiera.
- Organizar datos para acceso rápido y seguro.
- Manejar las comunicaciones en red. El Sistema Operativo permite al usuario manejar con alta facilidad todo lo referente a la instalación y uso de las redes de computadoras.
- Procesamiento por bytes de flujo a través del bus de datos.
- Facilitar las entradas y salidas. Un Sistema Operativo debe hacerle fácil al usuario el acceso y manejo de los dispositivos de Entrada/Salida de la computadora.

COMPONENTES DEL SISTEMA OPERATIVO

Un sistema operativo está conformado básicamente por cuatro módulos:

- **Núcleo o Kernel.**
- **Administrador de memoria.**
- **Sistema de entrada/salida.**
- **Administrador de archivos.**

A veces se considera un quinto módulo: el *intérprete de comandos* o *intérprete de instrucciones*, el cual se encarga de "traducir" las órdenes que el usuario ingresa mediante el teclado u otros dispositivos a un "lenguaje" que la máquina pueda entender.

Núcleo

Es el módulo de más bajo nivel de un sistema operativo, pues descansa directamente sobre el hardware de la computadora. Entre las tareas que desempeña se incluyen el manejo de las interrupciones, la asignación de trabajo al procesador y el proporcionar una vía de comunicación entre los distintos programas. En general, el núcleo se encarga de controlar el resto de los módulos y sincronizar su ejecución.

El núcleo contiene un submódulo denominado *"planificador"*, el cual se encarga de asignar tiempo del procesador a los programas, de acuerdo a una cierta política de planificación que varía de un sistema operativo a otro. Normalmente se utiliza una jerarquía de prioridades que determinan cómo se asignará el tiempo del CPU a cada programa. Una política de planificación muy común en los sistemas de multiprogramación y multiproceso son las técnicas de *"time slicing"* (fracción de tiempo). Se asigna a cada programa un corto intervalo de tiempo del procesador. Si el programa no ha terminado durante este intervalo de tiempo, vuelve a la cola de programas.

Administrador de memoria

Este módulo se encarga de asignar ciertas porciones de la memoria principal (RAM) a los diferentes programas o partes de los programas que la necesiten, mientras el resto de los datos y los programas se mantienen en los dispositivos de almacenamiento masivo. De este modo, cuando se asigna una parte de la memoria principal se hace de una forma estructurada, siguiendo un determinado orden. La forma más común de administración de la memoria supone crear una memoria virtual; con este sistema, la memoria de la computadora aparece, para cualquier usuario del sistema, mucho mayor de lo que en realidad es.

Sistema de entrada/salida (E/S)

Este componente presenta al usuario la E/S de datos como una cuestión independiente del dispositivo; es decir, para los usuarios, todos los dispositivos tienen las mismas características y son tratados de la misma forma, siendo el sistema operativo el encargado de atender las particularidades de cada uno de ellos (como su velocidad de operación). Una técnica muy común, especialmente en salida, es el uso de **"**spoolers"**.** Los datos de salida se almacenan de forma temporal en una cola situada en un dispositivo de almacenamiento masivo (el spool), hasta que el dispositivo periférico requerido se encuentre libre; de este modo se evita que un programa quede retenido porque el periférico no esté disponible. El sistema operativo dispone de llamadas para añadir y eliminar archivos del spool.

Administrador de archivos

Se encarga de mantener la estructura de los datos y los programas del sistema y de los diferentes usuarios (que se mantienen en archivos) y de asegurar el uso eficiente de los medios de almacenamiento masivo. El administrador de archivos también supervisa la creación, actualización y eliminación de los archivos, manteniendo un directorio con todos los archivos que existen en el sistema en cada momento y coopera con el módulo administrador de memoria durante las transferencias de datos desde y hacia la memoria principal. Si se dispone de un sistema de memoria virtual, existen transferencias entre la memoria principal y los medios de almacenamiento masivo para mantener la estructura de la misma.

Los archivos almacenados en los dispositivos de almacenamiento masivo tienen distintos propósitos. Algunos contienen información que puede ser compartida. Otros son de carácter privado, e incluso secreto. Por tanto, cada archivo está dotado de un conjunto de privilegios de acceso, que indican la extensión con la que se puede compartir la información contenida en el archivo. El sistema operativo comprueba que estos privilegios no sean violados.

FUNCIONES DEL KERNEL

El kernel o núcleo de Linux se puede definir como el corazón de este sistema operativo. Es el encargado de que el software y el hardware del computador puedan trabajen juntos. Las funciones del Kernel se simplifican en:

1. Administración de la memoria, para todos los programas en ejecución.
2. Administración del tiempo de procesador, que estos programas en ejecución utilizan.
3. Acceder a los periféricos/elementos y hardware de entrada y salida de una forma practica y cómoda.

El usuario de Linux puede adaptar el Kernel a sus necesidades configurando y compilando un nuevo Kernel o simplemente parchando y compilando el Kernel existente.

INTERPRETES DE ÓRDENES

En FreeBSD, gran parte del trabajo diario se realiza a través de un interfaz de la linea de órdenes llamado "shell". El principal trabajo del shell es recoger órdenes del canal de entrada y ejecutarlas. Muchos shells tienen también integradas funciones para ayudarnos en las tareas cotidianas tales como la manipulación de archivos, edición de líneas de órdenes, macros de órdenes, expansión de expresiones regulares en nombres de archivo y variables del sistema.

FreeBSD viene con un conjunto de shells, como sh, el shell Bourne y csh, el shell C. Hay disponibles muchos otros shells en la "FreeBSD Ports Collection" que son mucho más potentes, como bash y tcsh.

¿Qué shell usar? Es cuestión de gustos. Si se es programador de C se puede sentir más cómodo con tcsh, un shell con una sintaxis similar al C. Si se proviene del mundo Linux o se es nuevo en el interfaz de órdenes de Unix, se puede probar con bash. El asunto es que cada shell posee unas propiedades únicas que pueden o no funcionar con el entorno de trabajo preferido y se ha de efectuar una elección sobre el shell a usar.

Una de las propiedades comunes de un shell es que completa los nombres de archivo. Dada la introducción de las primeras letras de una orden o del nombre de un archivo, se puede hacer que el shell complete automáticamente el resto de la orden o el nombre del archivo pulsando la tecla TAB. Aquí va un ejemplo. Supongamos que se tienen dos archivos llamados foobar y foo.bar. Se quiere borrar foo.bar. Lo que habría que teclear es: rm fo[TAB].[TAB].
El shell nos mostraría rm foo[BEEP].bar.

El [BEEP] es el pitido de consola (*console bell*): es el shell diciéndonos que fue incapaz de completar totalmente el nombre de archivo porque hay más de una coincidencia. Tanto foobar como foo.bar comienzan por fo, pero solo se pudo completar hasta foo. Si se teclea ., y de nuevo TAB, el shell será capaz de introducir el resto del nombre por nosotros.

Otra función del shell son las variables de entorno. Las variables de entorno son parejas de valores clave almacenados en el espacio de entorno del shell. Este espacio puede ser leído por cualquier programa invocado por el shell y, por tanto, en él se encuentra bastante información relativa a la configuración de programas. Lo siguiente es una lista de las variables de entorno más comunes y su significado:

Variable	**Descripción**
USER	Nombre de usuario con el que se ha entrado al sistema.
PATH	Lista de directorios, separada por puntos y coma, en los que se busca ejecutables.
DISPLAY	Nombre en la red de la pantalla de X11 a la que conectarse, si se encuentra disponible.
SHELL	El shell actual.
TERM	El nombre del terminal del usuario. Se usa para determinar las posibilidades del terminal de datos.
TERMCAP	Base de datos donde encontrar los códigos de escape necesarios para realizar diferentes funciones en el terminal.
OSTYPE	Tipo de sistema operativo. Por ejemplo, FreeBSD.

Variable	Descripción
MACHTYPE	Arquitectura de la CPU en la que el sistema se está ejecutando.
EDITOR	El editor de texto preferido por el usuario.
PAGER	El paginador de texto preferido por el usuario.
MANPATH	Lista de directorios en los que se busca páginas de manual, separados por puntos y coma.

Visualizar o establecer una variable de entorno difiere ligeramente de shell a shell. Por ejemplo, en los shells al estilo C como tcsh y csh, se usaría setenv para establecer y visualizar las variables de entorno actuales. Siguiendo el ejemplo, para establecer o modificar el valor de EDITOR, bajo csh o tcsh una orden como la siguiente establecería el valor de EDITOR a /usr/local/bin/emacs:

% **setenv EDITOR /usr/local/bin/emacs**

Bajo los shells tipo Bourne (Bourne Shells):

% **export EDITOR="/usr/local/bin/emacs"**

También se puede hacer que la mayoría de los shells muestren el contenido de una variable de entorno situando el carácter $ delante del nombre de la variable desde la línea de órdenes. Por ejemplo, echo $TERM mostrará cualquiera que sea el valor que se le haya establecido a TERM, porque el shell expande el valor de TERM y se lo pasa al programa echo.

Los shells manejan muchos caracteres especiales, llamados meta-caracteres, como representaciones especiales de datos. El mas común es el carácter *, que representa cualquier número de caracteres en un nombre de archivo. Estos meta-caracteres especiales se pueden usar para la expansión de nombres de archivos. Por ejemplo, teclear echo * es casi lo mismo que introducir ls porque el shell recoge todos los archivos que coinciden con * y se los pone en la línea de órdenes a echo para que los vea.

Para evitar que el shell interprete estos caracteres especiales, se pueden salvar poniendo el carácter contrabarra (\) delante de ellos. echo $TERM muestra cualquiera que sea el valor establecido para el terminal que estamos usando. echo \$TERM muestra $TERM tal cual.

SISTEMA DE ARCHIVOS

La mayoría de los sistemas operativos poseen su propio sistema de archivos. Los sistemas de archivos son representados ya sea textual o gráficamente utilizando un gestor de archivos. Los sistemas de archivos más comunes utilizan dispositivos de almacenamiento de datos que permiten el acceso a los datos como una cadena de bloques de un mismo tamaño, a veces llamados sectores, usualmente de 512 bytes de longitud. El software del sistema de archivos es responsable de la organización de estos sectores en archivos y directorios y mantiene un registro de qué sectores pertenecen a qué archivos y cuáles no han sido utilizados. En la realidad, un sistema de archivos no requiere necesariamente de un dispositivo de almacenamiento de datos, sino que puede ser utilizado también para acceder a datos generados dinámicamente, como los recibidos a través de una conexión de red.

Generalmente un sistema de archivos tiene directorios que asocian nombres de archivos con archivos, usualmente conectando el nombre de archivo a un índice en una tabla de asignación de archivos de algún tipo, como FAT en sistemas de archivos MS-DOS o los inodos de los sistemas Unix. La estructura de directorios puede ser plana o jerárquica (ramificada o "en árbol"). En algunos sistemas de archivos los nombres de archivos son estructurados, con sintaxis especiales para extensiones de archivos y números de versión. En otros, los nombres de archivos son simplemente cadenas de texto y los metadatos de cada archivo son alojados separadamente.

En sistemas de archivos jerárquicos, en lo usual, se declara la ubicación precisa de un archivo con una cadena de texto llamada "ruta". La nomenclatura para rutas varía ligeramente de sistema en sistema, pero mantienen por lo general una misma estructura. Una ruta viene dada por una sucesión de nombres de directorios y subdirectorios, ordenados jerárquicamente de izquierda a derecha y separados por algún carácter especial que suele ser una barra ('/') o barra invertida ('\') y puede terminar en el nombre de un archivo presente en la última rama de directorios especificada.

Así, por ejemplo, en un sistema Unix la ruta a la canción preferida del usuario "pedro" sería algo como:
/home/pedro/música/Magic Mushrooms - Open Source.ogg
Un ejemplo análogo en un sistema de archivos Windows se vería como:
C:\Mis documentos\Mi música\Magic Mushrooms - Open Source.ogg

Los sistemas de archivos tradicionales proveen métodos para crear, mover y eliminar tanto archivos como directorios, pero carecen de métodos para crear, por ejemplo, enlaces adicionales a un directorio o archivo (enlace duro en Unix) o renombrar enlaces padres (".." en Unix).

El acceso seguro a sistemas de archivos básicos puede estar basado en los esquemas de lista de control de acceso o capacidades. Las listas de control de acceso hace décadas que demostraron ser inseguras, por lo que los sistemas operativos experimentales utilizan el acceso por capacidades. Los sistemas operativos comerciales aún funcionan con listas de control de acceso.

Los sistemas de archivos pueden ser clasificados en tres ramas: sistemas de archivos de disco, sistemas de archivos de red y sistemas de archivos de propósito especial.

Sistemas de archivos de disco

Un sistema de archivo de disco está diseñado para el almacenamiento de archivos en una unidad de disco, que puede estar conectada directa o indirectamente a la computadora.

Ejemplos de sistemas de archivos de disco:

EFSa
EXT2
EXT3
FAT (sistemas de archivos de DOS y Windows)
UMSDOS
FFS
Fossil
HFS (para Mac OS)
HPFS
ISO 9660 (sistema de archivos de solo lectura para CD-ROM)
JFS
kfs
MFS (para Mac OS)
Minix
NTFS (sistemas de archivos de Windows NT)
OFS
ReiserFS
Reiser4
UDF (usado en DVD y en algunos CD-ROM)
UFS
XFS

Algunos de estos son sistemas de archivos de registro por diario.

Sistemas de archivos de red

Un sistema de archivos de red es un sistema de archivos que accede a sus archivos a través de una red. Dentro de esta clasificación encontramos dos tipos de sistemas de archivos: los sistemas de ficheros distribuidos (no proporcionan E/S en paralelo) y los sistemas de ficheros paralelos (proporcionan una E/S de datos en paralelo).

Ejemplos de sistemas de archivos distribuidos pueden ser:

AFS
AppleShare
CIFS (también conocido como SMB o Samba)
Coda
InterMezzo
NSS (Para sistemas Novell Netware 5)
NFS
Mientras que para los sistemas de archivos paralelos tendríamos:
PVFS
PAFS

Sistemas de archivos de propósito especial

Los sistemas de archivos de propósito especial son básicamente aquellos que no caen en ninguna de las dos clasificaciones anteriores.

- acme (Plan 9)
- archfs
- cdfs
- cfs
- devfs
- udev
- ftpfs
- lnfs
- nntpfs
- plumber (Plan 9)
- procfs
- ROMFS
- Swap
- sysfs
- TMPFS
- wikifs

ARCHIVOS Y DIRECTORIOS

Los archivos y directorios están todavía organizados en notación DOS 8.3. Los nombres de archivos y carpetas están grabados en SBCS o MBCS (códigos de byte único y múltiple). La jerarquía máxima de 'profundidad' de una carpeta no puede exceder de 8 niveles.

INSTALACION DE WINDOWS

Copia los archivos *.CAB del CD de instalación de Windows a una carpeta en tu disco duro, pero al hacer esto cada vez que el sistema necesita estos archivos no los tomara automáticamente y te preguntara dónde están.

¿Cómo puedes hacer para que la carpeta donde están estos archivos sea la carpeta por default para que Windows localice los archivos de instalación?

Para lograr esto debes modificar el Registro de configuraciones. Debes darle un click a **inicio** y escribir la palabra **REGEDIT**, paso continuo ve a la clave:
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Setup
En la parte derecha de la ventana, busca la subclave "**SourcePath**". Haz un doble click en ella y en el cuadro que aparece escribe la dirección donde se encuentran los archivos **.CAB** en el disco duro.De esta manera ya Windows no te preguntará dónde se encuentran los archivos de instalación.

Cambio de la ruta para la Instalación de Windows

Si has instalado Windows 95 desde al unidad CD ROM, cuando quieras agregar nuevas aplicaciones de Windows, el sistema te pedirá el CD de instalación de windows en donde se encuentran los ficheros, para usar los CAB desde el dico duro tienes que hacer el siguiente cambio:

1. Ejecuta **Regedit** y busca la cadena: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Setup

2. Modifica la clave "**SourcePath**" D:\Win95 por C:\Win95 (Siendo D la unidad de CD_ROM).

INSTALACIÓN DE LINUX

El Linux funciona correctamente en un 386 SX con 4 Mb de memoria. Pero se deberá tener en cuenta que aquellas aplicaciones con fuerte requerimiento de memoria y carga de la CPU (por ejemplo, las X Windows) irán sumamente lentas. Una configuración mucho más utilizable sería la siguiente: 486 DX 33 MHz o superior, con 8 Mb de memoria RAM y con tarjeta de vídeo VGA o superior. Unidad de CD-ROM es altamente recomendable. Casi mejor habría que decir que imprescindible, dado que las distribuciones vienen en ese formato. En la actualidad, la mayoría de ellas están soportadas por el Linux. Asimismo tiene soporte para módem y tarjeta de sonido.

Linux corre tambien en Laptops o Notebooks 386 o superior, pudiendo correr también X-Windows en la mayoría de ellos. Existe una página donde se puede encontrar información: Linux Laptop.

En cuanto al espacio en disco duro, dependerá en gran medida de lo que se pretenda hacer desde Linux Con 100 Mb resulta espacio suficiente para un Linux completo, incluidas las X Windows. A partir de ahí, lo que cada uno considere.

Cómo instalar Linux**?**

- Para instalar Linux y hacer algo bueno, es recomendable 500 MB de espacio libre en el disco rígido.
- En todas las distribuciones de Linux viene un programa llamado FIPS, que se encarga de rearmar la tabla de particiones de nuestro disco para crear una nueva partición para albergar a Linux.
- Antes de utilizar FIPS, tenemos que defragmentar el disco, para tener toda la información junta y que nada se borre con FIPS.
- Luego, pasamos a crear 2 particiones: una para albergar a Linux, y una del mismo tamaño que la memoria, que servirá de swap (memoria virtual).
- Ahora con el CD de la distribución creamos los discos de inicio que contienen el programa de instalación. Esto depende de cada distribución.
- Metemos los discos de inicio y reiniciamos el sistema para arrancar el programa de instalación.

. A continuación algunas características de Linux y sus homólogos de otros sistemas.

	Linux	DOS	Windows 95/98	Windows NT	Novell
Requiere Hardware mínimo	no	no	si	si	si
Virus	no	si	si	si	si
Precio	US$ 0.-	-	US$ 60.-	US$ 3000.-	US$ 2500.-
Redes	si	no	no	si	si
Servidor de Internet	si	no	no	no	no

Linux desarrolla la mente de sus usuarios. Linux no se sujeta a la imposición y restricciones del software comercial. Cada uno de los miles de programas y aplicaciones son completamente gratuitos y están acompañados con el código fuente escrito por los programadores quienes crearon el sistema. Si algo no le agrada o si Ud. se ingenia una mejora puede aportar su idea al mundo Linux. He aquí algunas comparaciones con DOS, sistema operativo con el cual la mayoría de nosotros empezamos a conocer la computadora y que conforma el esqueleto de Windows:

- No existe el concepto de unidad de disco. Todas las unidades en Linux se 'montan' (se crean) como si fueran un subdirectorio más.
- No existe el concepto de extensión del nombre de un fichero. Los ficheros pueden tener nombres de hasta 256 caracteres. Los puntos están permitidos en el nombre de un fichero. Así, un fichero se podrá llamar: DOSEMU-HOWTO.español.tar.gz por poner un ejemplo.
- Los subdirectorios no se separan con el carácter '\', como en DOS, sino con el carácter '/'. Ejemplo: /usr/src/linux -1.2.13/Makefile
- Existe diferencia entre mayúsculas y minúsculas. Por ejemplo, no es lo mismo 'dir' que 'DIR' que 'Dir'...
- IMPORTANTE: Un sistema Linux NUNCA se puede apagar por las buenas. Antes le hemos de advertir al S.O. de que vamos a apagarlo (o reiniciarlo). La razón de que esto deba ser así es para que al sistema le dé tiempo de escribir en disco todos los datos que tuviera pendientes de escribir, salir ordenadamente de todas las aplicaciones que tuviera arrancadas y desmontar todas las unidades que tuviera montadas.

Antes que nada conviene aclarar que Linux puede convivir con otros sistemas operativos en la misma maquina, es decir, puede correr Windows o DOS juntamente con Linux.

REQUERIMIENTO PARA INSTALAR LINUX

- **CPU**: PC con procesador 386, 486, Pentium, Pentium Pro, Pentium II o unos de los clones de estos procesadores hechos por fabricantes como Cyrix, AMD, TI, IBM; etc. El sistema no funcionará en un 286 o en procesadores inferiores.
- **Bus de E/S**: debe utilizar el bus ISA, EISA, PCI o VL. El bus VL se conoce también como VESA Local Bus o VLB. Las computadoras que tienen PCI o VLB generalmente tienen también slots ISA o EISA.
- **RAM y Disco Rígido**: debe tener al menos 4 MB de RAM, aunque es recomendable no menos de 16 MB y 40 MB de disco duro. Pero si queremos instalar todo, necesitaremos 400 MB. Las interfaces de disco que emulan la interfaz de disco "AT" –llamadas habitualmente MEN, RLL, IDE o ATA- están soportadas. Las controladoras de disco SCSI de diversos fabricantes también están soportadas.
- **Unidad de Disquete**: el sistema en el que se instala LINUX debe tener una unidad de disquete de 1,44MB.
- **MPC:** como casi todas las distribuciones de LINUX vienen en CD-ROM, se necesita una lectora que puede ser IDE, SCSI o norma propia, como por ejemplo Sanyo.
- **Placa de Vídeo**: es recomendable usar una placa de vídeo compatible con VGA para la terminal de la consola. Prácticamente cualquier tarjeta gráfica moderna es compatible con VGA, CGA, MDA o HGA pueden funcionar correctamente para texto, pero no funcionaran con el X Windows.
- **Otro Hardware**: LINUX soporta una gran variedad dispositivos, como mouse, impresora, escáner, módem, tarjeta de red, etc. Sin embargo, no se requiere ninguno de estos dispositivos durante la instalación del sistema.

INSTALACIÓN UNIX

El ``Esquema de prefijo'' se utiliza cuando se desea utilizar una instalación de Pyhton para realizar la construcción/instalación (por ejemplo, ejecutar el guion de instalación o setup), pero la instalación de los módulos en el directorio de instalación para módulos generados por terceros, en un directorio distinto del de la instalación de Python (o que luzca diferente a un directorio de instalación de Pyhton). Si esto le parece un tanto inusual, es por que realmente lo es, por esa razón el ``esquema home'' siempre va de primero. Sin embargo, hay al menos dos casos conocidos donde el esquema de prefijo es de mucha utilidad.

Primero, considere que en muchas distribuciones de Linux, Python se instala en /usr, a diferencia del tradicional /usr/local. Esto resulta apropiado si se considera que Python se instala como parte del ``sistema'' a diferencia de una adicion extra. Sin embargo cuando se instala módulos de Python a partir de una distribución, probablemente se desee que estos se instalaen en /usr/local/lib/python1.X en vez de /usr/lib/python1.X. This can be done with
/usr/bin/python setup.py install --prefix=/usr/local

Otra posibilidad es, en un sistema de redes de archivos, donde el nombre para escritura en un directorio remoto difiere del nombre para lectura, por ejemplo, el interpretador de Python accesado en /usr/local/bin/python buscaría los módulos en /usr/local/lib/python1.X, pero estos módulos tendrian que ser instalados en /mnt/@server/export/lib/python1.X. Esto pudiera lograrse utilizando
/usr/local/bin/python setup.py install --prefix=/mnt/@server/export

De cualquier manera, la opción **--prefix** define la instalación base específica para cada plataforma, utilizada por archivos específicos de cada plataforma. (En la actualidad esto implica aquellas distribuciones de módulos ``no puras'', pero se puede expandir a librerias de C, ejecutables binarios,etc.) Si no se asigna **--exec-prefix**, se toma por defecto a **--prefix**. Los archivos se instalan como sigue:

Tipo de archivo	**Directorio de Instalación**	**opcion Override**
distribución pura de módulo	prefix/lib/python1.X/site-packages	**--install-purelib**
distribución de módulo (no pura)	exec-prefix/lib/python1.X/site-packages	**--install-platlib**
scripts	prefix/bin	**--install-scripts**
data	prefix/share	**--install-data**

No hay ningún requerimiento para que **--prefix** o **--exec-prefix** apunten a una instalación alterna de Python. Si los mencionados directorios no existen, estos so creados al momento de la instalación.

Incidentalmente la verdadera razón de la importancia del esquemo de prefijo es que las instalaciones estándar bajo Unix utilizan dicho esquema pero con **--prefix** y **--exec-prefix** dados por Python (como sys.prefix y sys.exec_prefix). En consecuencia se podria pensar que no hay ninguna otra necesidad de utilizar el esquema de prefijo, pero la verdad es que cada vez que se ejecuta python setup.py install sin mas opciones esta haciendo uso de ello.

Note que el instalar extensiones en una instalación alterna de Python no causa ningún efecto en como esas extensiones han sido construidas: en particular, los archivos de cabecera de Python (Python.h y conocidos) con el interpretador de Pyhton utilizado para ejecutar el guion de inicio o setup, siendo utilizado para compilar las extensiones. Es su responsabilidad la de asegurarse que el interpretador para ejecutar las extensiones instaladas de esta manera, es compatible con el interpretador utilizado en su construcción y que ambos interpretadores son de la misma version de Python (posiblemente diferentes construcciones o posiblemente copias de la misma construcción). (por supuesto, si su **--prefix** y **--exec-prefix** ni siquiera apuntan a una instalación alterna de Pyhton, esto es totalmente irrelevante.)

PARTICION DE UN DISCO DURO

Es peculiaridad de cada Sistema Operativo establecer una norma para el almacenamiento de archivos en el disco duro. Cuando solo usábamos MS-DOS el disco duro era por lo general utilizado en su totalidad, no solía ser necesario el particionado del mismo. Con la aparición de otros sistemas operativos y el tamaño de los nuevos discos duros, esto, se ha hecho necesario.

Efectuar la partición de un disco duro es un proceso distinto a su formateo. Cuando realizamos la partición del disco duro, indicamos qué secciones del mismo serán utilizadas por DOS o por el otro sistema operativo. Al formatear, MS-DOS prepara cada partición para recibir archivos y será necesario formatear cada una de las particiones efectuadas.

Las particiones que pueden efectuarse en el disco duro pueden ser de dos tipos: Partición primaria de DOS y Partición extendida de DOS.

PARTICION PRIMARIA DE DOS (SOLO EN W9X / ME)

Al iniciar el arranque, el disco duro deberá tener una partición primaria de DOS que contenga los tres archivos del sistema: IO.SYS, MSDOS.SYS y COMMAND.COM, y dicha partición debe corresponder a la partición activa.

Por lo general a esta partición primaria del primer disco se le asigna la letra C. Como ya hemos planteado, es posible reservar una porción del disco para esta partición y dejar el resto para otras particiones.

PARTICIÓN EXTENDIDA DEL DOS

Del espacio de disco no utilizada para la partición primaria, se pueden crear otras particiones llamadas Extendidas y estas a su vez se pueden dividir en una o más unidades lógicas, siendo 26 las letras disponibles para estas unidades, desde la A a la Z.

Las unidades A y B se reservan para disquetes y la C generalmente para la partición primaria de DOS.

PARTICIÓN NO DOS

Las particiones NO-DOS son las utilizadas por otros sistemas operativos, no siendo posible crearlas con la utilidad FDISK de la versión DOS.

PARTICIÓN ACTIVA

Un disco duro sólo podrá tener una partición activa en un tiempo dado, para iniciar un sistema operativo desde el disco duro es necesario activar la partición primaria donde se encuentra el Sistema Operativo.

FORMATEO DE DISCO

Luego de crear la partición primaria dos, tenemos que formatear el disco para lo cual ponemos el diskette de inicio o booteable, una vez iniciado y estando en a: tipeamos **format c:** donde c: es la unidad que queremos formatear, nos avisa que vamos a perder los datos de la unidad y si realmente queremos formatear, ponemos que si.

Luego hace un pequeño y rápido conteo donde nos indica que esta guardando los datos de reconstruir, y empieza el formato donde va a contar del 1 al 100% del disco.

Si mientras esta contando el formateo nos aparece un mensaje informativo donde nos dice que se esta tratando de recuperar unidad de asignacion, deberemos reemplazar el disco por otro en buenas condiciones, sin sectores dañados.

Una vez llegado al 100% nos va a preguntar un nombre de volumen, le ponemos el nombre que queramos o nada y listo ya esta formateado.

Ya podemos cargar el sistema operativo que nos convenga y hallamos elegido.

Diferentes comandos de format

Format c:/q formato rápido. (únicamente si el disco ya esta formateado)

Format c:/u no guarda la información de reconstrucción (es el mas recomendable)

Format c:/s carga los archivos de sistemas

CARACTERISTCIAS DE UNIX

Las características principales de este SO es que se diseñó como un sistema de tiempo compartido. Esto quiere decir que varios usuarios estarán usando el sistema al mismo tiempo, sin notar la presencia de los demás en su ámbito de trabajo, lo que logra el SO asignándole un tiempo de atención para el CPU a cada usuario.

La interfaz estándar con el usuario (el *shell*) puede ser cambiada si se quiere. La mayoría del código fuente original está disponible por lo que los usuarios podrán ajustar el SO a sus requerimientos específicos. Es multitarea, es decir que permite que se puedan ejecutar varios procesos al mismo tiempo compartiendo el uso de la CPU.

Soporta el procesamiento en tiempo real (ejecución de procesos en intervalos de tiempo especificados sin retardo), el cual se utiliza en aplicaciones de robótica y base de datos.

Los sistemas UNIX son esenciales para la Internet.

Los sistemas de archivos con árboles multiniveles permiten que el SO trate tanto a directorios y archivos como simples secuencias de bytes.

Posee distintos niveles de seguridad: password's de ingreso y permisos de archivos y directorios.

Un proceso puede fácilmente generar otro, también es posible el manejo de procesos en determinado tiempo. Puede planearse la utilización de la CPU.

El kernel y biblioteca del SO están preparados para que el SO pueda extenderse y crecer, lo que permitió a UNIX mantenerse siempre a la cabeza de los SO's estando permanentemente actualizado.

SISTEMA OPERATIVO WINDOWS

Microsoft es el gigante informático que produce y comercializa Windows, el sistema operativo que usa el 90% de los ordenadores personales de todo el mundo (su última versión es Windows XP). Su precio ronda los 120 euros y muchas veces se incluye en el precio del propio ordenador (que ronda los 1.200 euros).

Al igual que un turismo sirve para moverse por casi todo tipo de carreteras, Windows es el estándar de facto que cubre la gran mayoría de necesidades del usuario medio. Ya sea para escribir documentos, navegar por Internet, escuchar música, ver películas, retocar fotografías digitales o disfrutar de los últimos juegos, Windows es fácil de usar y configurar, sin necesidad de poseer conocimientos informáticos avanzados. Además, la práctica totalidad de los programas que se comercializan disponen de una versión para Windows. Sin embargo, Windows tiene fama de ser inestable, ya que los bloqueos y cuelgues son frecuentes (sobre todo en versiones anteriores a XP). Su seguridad también deja mucho que desear, pues existen multitud de virus que aprovechan fallos del sistema para infectar el PC, como, por ejemplo, el virus Sasser o el más dañino Chernobyl.

SISTEMA OPERATIVO LINUX

Es el sistema preferido por muchos de los profesionales de la informática y de Internet. Como si de un 4x4 se tratase, ofrece potencia, estabilidad, seguridad contra virus y sirve para realizar cualquier trabajo. Por contra, al igual que un todo terreno, peca de falta de comodidad, no es tan fácil de usar como los otros dos y se necesitan conocimientos técnicos para realizar algunas tareas. Aún así, sus últimas versiones son bastante más amigables, por lo que ha empezado a calar entre algunos usuarios domésticos y ofimáticos (uso del ordenador en ambientes de oficina).

La colección de programas disponibles en algunas áreas (como los juegos o las aplicaciones multimedia) es aún escasa o de poca calidad, aunque mejora día a día. A la hora de instalar nuevos periféricos, el proceso también se complica: mientras que en Windows y Mac OS es casi automático, en Linux suele ser necesario realizar alguna configuración manual.

Linux es un sistema un tanto 'romántico', pues ha sido desarrollado voluntariamente por programadores de todo el mundo. De ahí que sea libre (libertad total para ver y modificar las 'entrañas' del sistema) y gratuito

Qué es un proceso [DEIT93] [LIST86] [TANE93]

> Hasta ahora hemos utilizado siempre el término programa. A partir de ahora distinguiremos entre programa y proceso. Un programa es una secuencia de instrucciones escrita en un lenguaje dado. Un **proceso** es una instancia de ejecución de un programa, caracterizado por su contador de programa, su palabra de estado, sus registros del procesador, su segmento de texto, pila y datos, etc. Un programa es un concepto estático, mientras que un proceso es un concepto dinámico. Es posible que un programa sea ejecutado por varios usuarios en un sistema multiusuario, por cada una de estas ejecuciones existirá un proceso, con su contador de programa, registros, etc. El sistema operativo necesita el concepto de proceso para poder gestionar el procesador mediante la técnica de multiprogramación o de tiempo compartido, de hecho, el proceso es la unidad planificable, o de asignación de la CPU.

Estados de un proceso y Transiciones de estado de los procesos [DEIT93] [TANE93] [SILB94][STAL95]

Durante su vida, un proceso puede pasar por una serie de estados discretos, algunos de ellos son:

- **En ejecución**: El proceso ocupa la CPU actualmente, es decir, se está ejecutando.
- **Listo** o **preparado**: El proceso dispone de todos los recursos para su ejecución, sólo le falta la CPU.
- **Bloqueado:** Al proceso le falta algún recurso para poder seguir ejecutándose, además de la CPU. Por recurso se pueden entender un dispositivo, un dato, etc. El proceso necesita que ocurra algún **evento** que le permita poder proseguir su ejecución.

Hay otros estados de los procesos, pero en la presente exposición se tratarán estos tres. Por sencillez, se considera un sistema con una sola CPU, aunque no es difícil la extensión a múltiples **procesadores**. Solamente puede haber un proceso en ejecución a la vez, pero pueden existir varios listos y varios pueden estar bloqueados. Así pues, se forman una lista de procesos listos y otra de procesos bloqueados. La lista de procesos listos se ordena por **prioridad**, de manera que el siguiente proceso que reciba la CPU será el primero de la lista. La lista de procesos bloqueados normalmente no está ordenada; los procesos no se desbloquean (es decir, no pasan a ser procesos listos) en orden de prioridad, sino que lo hacen en el orden de ocurrencia de los eventos que están esperando. Como se verá más adelante, hay situaciones en las cuales varios procesos pueden bloquearse esperando la ocurrencia del mismo evento; en tales casos es común asignar prioridades a los procesos que esperan.

Transiciones de estado de los procesos

A continuación se dan ejemplos de eventos que pueden provocar transiciones de estado en un proceso en este modelo de tres estados (ver *figura 2.1*). La mayoría de estos eventos se discutirán con profundidad a lo largo del curso:

- ☑ **De ejecución á Bloqueado**: al iniciar una operación de E/S, al realizar una operación **WAIT** sobre un **semáforo** a cero (en el tema de procesos concurrentes se estudiarán los ***semáforos***).
- ☑ **De ejecución á Listo**: por ejemplo, en un sistema de **tiempo compartido**, cuando el proceso que ocupa la CPU lleva demasiado tiempo ejecutándose continuamente (agota su **cuanto**) el sistema operativo decide que otro proceso ocupe la CPU, pasando el proceso que ocupaba la CPU a estado listo.

☑ **De Listo á en ejecución**: cuando lo requiere el planificador de la CPU (veremos el planificador de la CPU en el tema de planificación de procesos).

☑ **De Bloqueado á Listo**: se dispone del recurso por el que se había bloqueado el proceso. Por ejemplo, termina la operación de E/S, o se produce una operación **SIGNAL** sobre el semáforo en que se bloqueó el proceso, no habiendo otros procesos bloqueados en el semáforo.

Niveles de Planificación [DEIT93] [MILE94] [STAL95]

La planificación de la CPU, en el sentido de conmutarla entre los distintos procesos, es una de las funciones del sistema operativo. Este despacho es llevado a cabo por un pequeño programa llamado **planificador a corto plazo** o ***dispatcher*** (*despachador*). La misión del *dispatcher* consiste en asignar la CPU a uno de los procesos ejecutables del sistema, para ello sigue un determinado algoritmo. En secciones posteriores estudiaremos algunos algoritmos posibles. Para que el *dispatcher* conmute el procesador entre dos procesos es necesario realizar un ***cambio de proceso***.

Los acontecimientos que pueden provocar la llamada al *dispatcher* dependen del sistema (son un subconjunto de las interrupciones), pero son alguno de estos:

- El proceso en ejecución acaba su ejecución o no puede seguir ejecutándose (por una E/S, operación **WAIT**, etc).
- Un elemento del sistema operativo ordena el bloqueo del proceso en ejecución (ver ***estados de un proceso***).
- El proceso en ejecución agota su **cuantum o cuanto** de estancia en la CPU.
- Un proceso pasa a estado listo.

Hay que destacar el hecho de que cuanto menos se llame al *dispatcher* menos tiempo ocupa la CPU un programa del sistema operativo, y, por tanto, se dedica más tiempo a los procesos del usuario (un cambio de proceso lleva bastante tiempo).

Así, si sólo se activa el *dispatcher* como consecuencia de los 2 primeros acontecimientos se estará haciendo un buen uso del procesador. Este criterio es acertado en sistemas por lotes en los que los programas no son interactivos. Sin embargo, en un sistema de tiempo compartido no es adecuado, pues un proceso que se dedicara a realizar cálculos, y no realizara E/S, monopolizaría el uso de la CPU. En estos sistemas hay que tener en cuenta el conjunto de todos los procesos, activándose el *dispatcher* con la circunstancia tercera y, posiblemente, la cuarta. Los sistema operativos en que las dos siguientes circunstancias no provocan la activación del dispatcher muestran preferencia por el proceso en ejecución, si no ocurre esto se tiene más en cuenta el conjunto de todos los

PROCESOS.

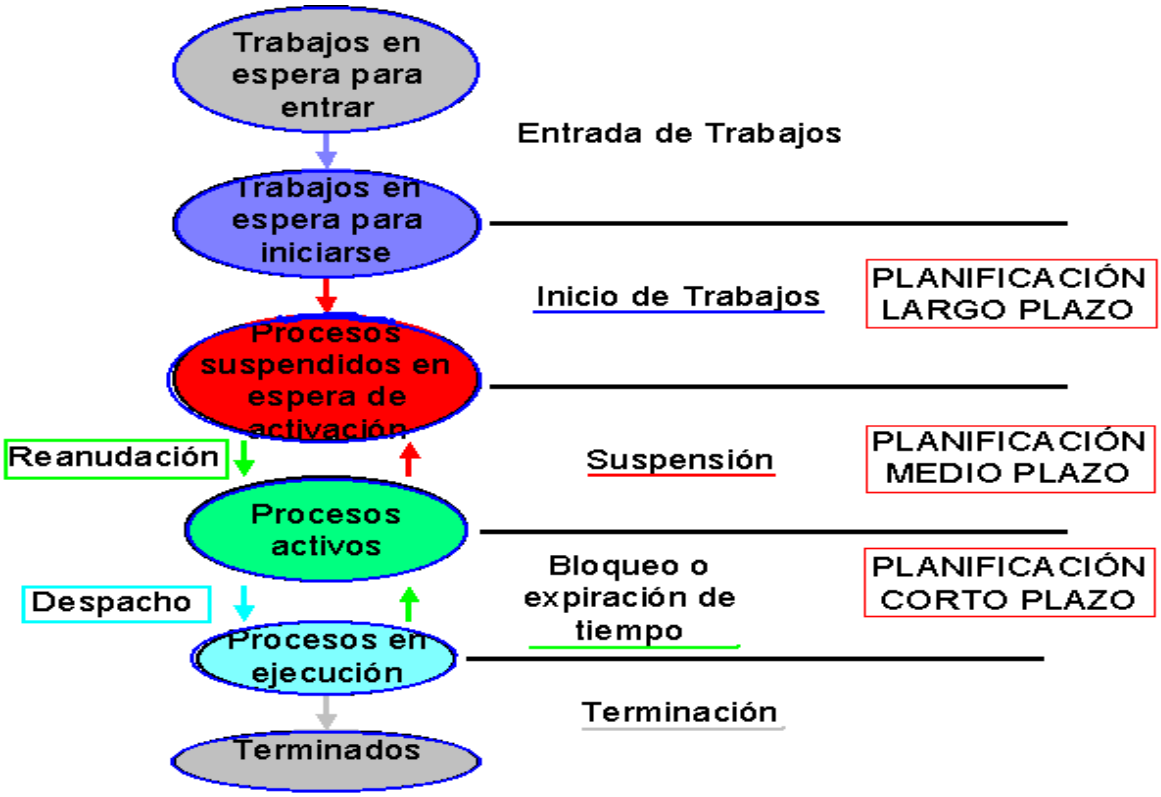


Figura1. Niveles de Planificación

Se puede definir el ***scheduling*** -algunas veces traducido como -**planificación**- como el conjunto de políticas y mecanismos construidos dentro del sistema operativo que gobiernan la forma de conseguir que los procesos a ejecutar lleguen a ejecutarse.

El *scheduling* está asociado a las cuestiones de:

- Cuándo introducir un nuevo proceso en el Sistema.
- Determinar el *orden de ejecución* de los procesos del sistema.

El *scheduling* está muy relacionado con la gestión de los recursos. Existen tres niveles de *scheduling*, como se ilustra en la figura 1.1, estos niveles son:

- Planificador de la CPU o a corto plazo.
- Planificador a medio plazo.
- *Planificador a largo plazo*.

Ya hemos hablado del planificador de la CPU, y en los subapartados posteriores se comentan los dos restantes:

PLANIFICACIÓN A LARGO PLAZO

Este planificador está presente en algunos sistemas que admiten además de procesos **interactivos** trabajos por **lotes**. Usualmente , se les asigna una prioridad baja a los trabajos por lotes, utilizándose estos para mantener ocupados a los recursos del sistema durante períodos de baja actividad de los procesos interactivos. Normalmente, los trabajos por lotes realizan tareas rutinarias como el cálculo de nóminas; en este tipo de tareas el programador puede estimar su gasto en recursos, indicándoselo al sistema. Esto facilita el funcionamiento del planificador a largo plazo.

El objetivo primordial del planificador a largo plazo es el de dar al planificador de la CPU una mezcla equilibrada de trabajos, tales como los limitados por la CPU (utilizan mucho la CPU) o la E/S. Así, por ejemplo, cuando la utilización de la CPU es baja, el planificador puede admitir más trabajos para aumentar el número de procesos listos y, con ello, la probabilidad de tener algún trabajo útil en espera de que se le asigne la CPU. A la inversa, cuando la utilización de la CPU llega a ser alta, y el **tiempo de respuesta** comienza a reflejarlo, el planificador a largo plazo puede optar por reducir la frecuencia de admisión de trabajos.

Normalmente, se invoca al planificador a largo plazo siempre que un proceso termina. La frecuencia de invocación depende, pues, de la carga del sistema, pero generalmente es mucho menor que la de los otros dos planificadores. Esta baja frecuencia de uso hace que este planificador pueda permitirse utilizar algoritmos complejos, basados en las estimaciones de los nuevos trabajos.

PLANIFICACIÓN A MEDIO PLAZO

En los sistemas de ***multiprogramación y tiempo compartido*** varios procesos residen en la memoria principal. El tamaño limitado de ésta hace que el número de procesos que residen en ella sea finito. Puede ocurrir que todos los procesos en memoria estén bloqueados, desperdiciándose así la CPU. En algunos sistemas se intercambian procesos enteros (*swap*) entre memoria principal y memoria secundaria (normalmente discos), con esto se aumenta el número de procesos, y, por tanto, la probabilidad de una mayor utilización de la CPU.

El planificador a medio plazo es el encargado de regir las ***transiciones de procesos*** entre memoria principal y secundaria, actúa intentando maximizar la utilización de los recursos. Por ejemplo, transfiriendo siempre a memoria secundaria procesos bloqueados, o transfiriendo a memoria principal procesos bloqueados únicamente por no tener memoria.

Creación de procesos: fork()

Los procesos de un sistema UNIX tienen una estructura jerárquica, de manera que un proceso (proceso padre) puede crear un nuevo proceso (proceso hijo) y así sucesivamente. Para la realización de aplicaciones con varios procesos, el sistema operativo UNIX proporciona la llamada al sistema[1] fork().

Cabecera:

```
#include <unistd.h>

int fork(void);
```

Comportamiento de la llamada:

fork() crea un nuevo proceso exactamente igual (mismo código) al proceso que invoca la función. Ambos procesos continúan su ejecución tras la llamada fork(). En caso de error, la función devuelve el valor **-1** y no se crea el proceso hijo. Si no hubo ningún error, el proceso padre (que realizó la llamada) obtiene el **pid** del proceso hijo que acaba de *nacer*, y el proceso hijo recibe el valor **0**.

Ejemplo:

En el ejemplo que se muestra a continuación, se crea un proceso hijo que imprime en pantalla el **pid** de su proceso padre, mientras que el proceso padre imprime en pantalla su propio **pid** y el del proceso hijo que ha creado. Para ello, se utilizan las llamadas al sistema getpid() y getppid(). El proceso padre, antes de finalizar se suspende hasta que el hijo acaba, para evitar que éste se quede *zombie*. Para ello, utiliza la llamada al sistema wait(), que recibe en la variable *status* el estado en que el proceso hijo finalizó.

```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>

int main() {

  int pid = 0, status = 0;

  if ((pid = fork()) == -1) {
    printf(``Error al crear proceso hijo\n'');
    exit(1);
  }
  if (pid == 0) { /* Proceso Hijo */
    printf(``El PID de mi proceso padre es %d\n'', getppid());
    exit(1);
  }
  else { /* Proceso Padre */
    printf(``Mi PID es el %d y he creado un proceso hijo cuyo pid es %d\n'',
           getpid(), pid);
    wait(&status);
    printf(``\nEl proceso hijo finalizo con el estado %d\n'', status);
    exit(0);
  }
}
```

INTRODUCCIÓN GESTION DE LA MEMORIA

La Memoria Principal es un recurso muy importante que se ha de gestionar, porque se ha de disponer de velocidad.

Hay dos tipos de tiempos cuando hablamos de memoria:

- Tiempo de acceso Tiempo de finalización de una petición menos el tiempo de inicio de la petición
- Tiempo de ciclo de la memoria Desde que finaliza una petición hasta que se inicia la siguiente petición. Este se ve detenidamente por el Hardware.
- **DIRECCIONAMIENTO**

Cuando escribimos un programa no indicamos el direccionamiento (donde guardamos el programa). Este se define solamente cuando comenzamos a escribir (donde pone la primera sentencia del programa). La primera línea de código marcará el momento de compilar la dirección de memora 0 relativa. De esta manera los programas son portables por diferentes sistemas y máquinas, separando la carga del programa en memoria de su ejecución

La primera sentencia sería la 0 relativo, seguidas de la 1 relativo,...

Cuando se coge la primera línea del programa para compilar y se pone la primera línea con 0 negativo. Y así el programa será + fácil para ejecutarse.

- **MONOPROGRAMACIÓN**

Con la llegada de la monoprogramación y de los sistemas operativos se hace necesario gestionar la memoria de manera que los programas que se querían cargar a la memora no pisen (borren,...) la memoria reservada al sistema operativo

MSDOS, cuenta con una ayuda que se ejecuta con el comando HELP. Lo malo es que hay versiones que no incorpora tal ayuda. La versión del WINDOWS no lo incorpora.

También podemos obtener ayuda de un comando determinado, introduciendo el comando y a continuación /?. Por ejemplo: dir /? Esto nos sacara la ayuda del DOS sobre este comando.

- **COMANDOS DE MS-DOS**

Comandos para la gestión de directorios

Comando: MD

Etimología: Viene de make directory.

Función: Crear directorios

Sintaxis: MD [unidad\ruta\]<nombre>

Comando: **RD**

Etimología: Viene de remove directory.

Función: Borra un directorio (solo si este se encuentra vació).

Sintaxis: RD [unidad\ruta\]<nombre>

Comando: DELTREE

Etimología: Anglicismo delete “eliminar/borrar” tree “árbol”.

Función: Borrar directorios (estando o no estando vacíos).

Sintaxis: DELTREE [unidad\ruta\]<nombre>

Comando: DIR

Etimología: De directorio ☺.

Función: Este comando procesa una visualización de todos los ficheros, directorios, de la ruta en la que nos encontramos.

Mediante una serie de parámetros podemos modificar ese listado de visualización. Este comando funciona como un filtro.

Sintaxis: DIR [unidad\directorio\fichero]

Parámetros: podemos especificar unos parámetros para que se listen los archivos y directorios de una forma concreta:

/P Con este parámetro podemos restringir la secuencia de listado y detenerla hasta que pulsemos una tecla. Al pulsar una tecla se procesara el siguiente bloque de listado y así sucesivamente. Este comando reparte internamente el numero de ficheros y directorios en bloques para luego ir sacándolos.

/N Ordena por nombre

/E Ordena por extensión

/S Ordena por tamaño

/D Ordena por fecha

/G Ordena poniendo agrupados todos los directorios después de los ficheros.

Comando: CD

Etimología: De change dir (cambiar directorio)

Función: Permite cambiar de un directorio activo a otro. Es por el cual podemos movernos en MDDOS.

Sintaxis: CD [unidad:]\[ruta]\[directorio]

Peculiaridades: **Si deseamos retroceder un directorio no hace falta poner la ruta nos basta con poner** cd..

Comando: TREE

Etimología: Palabra inglesa tree (árbol).

Función: Este comando nos mostrara la estructura o el árbol de directorios de la unidad especificada en función opcional de unos parámetros dados.

Sintaxis: tree [unidad\ruta] [/f] [/a]

Donde unidad/ruta podremos especificar la ruta de la cual deseamos obtener esta información. El parámetro /f hará que se muestren también los archivos dentro de esta estructura de directorios.

Comando: MOVE

Etimología: Viene de move (mover)

Función: Este comando mueve ficheros de un directorio a otro. Este comando vendría hacer internamente una copia del archivo al directorio especificado a mover, luego borra el fichero de salida.

Sintaxis: move [/y] <origen> <destino>

Donde /y es un parámetro que poniéndolo el move moverá ficheros sin preguntar la confirmación de reemplazo a otros archivos que se puedan llamar de la misma forma al directorio de destino. En caso de no especificarse, MSDOS nos preguntara la confirmación de reemplazo de ficheros. A continuación debemos especificar el directorio de origen y el de salida o destino.

Comandos para la gestión de archivos

Comando: TYPE

Etimología: Pos lo de siempre, anglicismo.

Función: **Ver el contenido de archivos de texto, haciendo un listado (no permite el uso de comodines).**
Sintaxis: **TYPE [unidad:]\[ruta]\[directorio]\<archivo>**

Comando: **PRINT**

Etimología: Pues de imprimir ☺

Función: Imprime archivo de texto.

Sintaxis: : **PRINT [unidad:]\[ruta]\[directorio]\<archivo>**

Comando: COPY

Etimología: Pos lo de siempre copy en el ingles de copiar.

Función: Este comando permite la reproducción o copia de archivos o ficheros (es lo mismo).

Sintaxis: copy <fichero-origen> <fichero-destino>

Comando: MOVE

Función: Este comando mueve ficheros de un directorio a otro. Este comando vendría hacer internamente una copia del archivo al directorio especificado a mover, luego borra el fichero de salida.

Sintaxis: move [/y] <origen> <destino>

Comando: REN o RENAME

Etimología: Viene de la palabra inglesa rename (renombrar).

Función: Su función es dar un nuevo nombre a un fichero. No se puede renombrar y que haya dos archivos con el mismo nombre, ni MSDOS ni Windows lo permiten dentro de un mismo directorio. Si los archivos llamados de forma igual se encuentran en directorios distintos si lo permiten. También podemos usar en este comando los comodines del dir ☺.

Sintaxis: rename <nombre-actual> <nombre-nuevo>

Comando: DEL

Etimología: Su nombre viene como siempre del ingles (delete) ☹.

Función: Es el comando encargado de eliminar archivos. En este comando podemos usar los comodines que usábamos en el comando dir.

Sintaxis: **del [unidad:]\[ruta]\[directorio]\<archivo>**

Comando: ERASE

Función: Exactamente igual que el anterior, por ello no le daré mas importancia.

Sintaxis**: erase [unidad:]\[ruta]\[directorio]\<archivo>**

Comando: FC

Etimología: Como siempre referencia al ingles (normal en la informática ☹) file compare.

Función: Sirve para comparar dos ficheros y verificar así que las copias son exactas.

Sintaxis: FC fichero1 fichero2

Comando: UNDELETE

Etimología: UN (recupera) DELETE (el borrado), traducción chapucera by Quasi.

Función: Permite recuperar ficheros previamente borrados con la orden DEL. Posee tres parámetros /LIST (presenta una lista de archivos recuperables), /ALL (recupera todos los archivos sin pedir confirmaciones) y /DOS (crea una lista de archivos borrados por el dos y otra de archivos borrados por otro motivo).

Sintaxis: UBDELETE [parámetros]<fichero>

Comando: XCOPY

Función: Con el COPY solo copiábamos archivos con este comando podemos copiar un directorio entero, con sus subdirectorios y archivos.

Sintaxis: xcopy origen [destino]

Donde origen es la ruta del directorio o archivo a copiar.

Parámetros: En este comando puedes incluir unos parámetros o modificadores que tendrá en cuenta a la hora de ser ejecutado, estos son:

/P : Pide confirmación de SI o NO antes de copiar cada archivo.

/S : Indica que la copia debe hacerse extensiva a todos los subdirectorios.

/E : Copia también los directorios vacíos.

/W : Espera la confirmación antes de copiar los archivos.

/V : Verifica la corrección de la copia.

/D Obliga a hacer una copia selectiva solo de los ficheros de fecha igual o mayor a la especificada.

Comando: DOSKEY

Función: Este comando se encuentra residente en memoria y una vez activado, permite visualizar los comandos de MSDOS introducidos con anterioridad.

Comando: VERIFY

Función: Comando que indica al sistema que archivos son escritos correctamente.

Sintaxis: verify (on/off)

Comando: EDIT

Función: En MSDOS podemos contar con un editor de texto, con una interface grafica para msdos ☺. Con este editor de texto implementado en MSDOS podemos editar cualquier archivo que contenga texto.

Sintaxis: edit [unidad:]\[ruta]\[directorio]\<archivo.ext (solo de texto)>

Ejemplo: c:\>edit autoexec.bat

Con esto se nos abriría el editor de texto del MSDOS con el archivo de autoexec.bat listo para editar, no metais la pezuña en este archivo si no conocemos su funcionamiento ☺.

En caso de que el nombre de archivo pasado como parámetro al edit no exista, el edit abrirá un archivo en blanco almacenado con ese nombre pero de forma temporal. Así que si queréis tenerlo, debéis guardarlo con el edit en un directorio.

Comandos para gestiones en discos y disquetes

Comando: FORMAT

Etimología: Viene de formatear, anglicismo.

Función: Formateado, o borrado completo de un disco o disquete.

Sintaxis: format <unidad:>

Parámetros: Permite los siguientes modificadores

/Q: Realiza formato rápido.

/Q y /U : El uso conjunto de estos dos parámetros asegura un formateado muy rápido.

Peculiaridades: También existe para recuperar lo formateado el UNFORMAT.

Comando: SYS

Etimología: De system (sistema).

Función: Crear un disco de sistema (un disco con los archivos de sistema)

Sintaxis: sys unidad:

Comando: CHKDSK

Etimología: Deriva de las palabras (check disck), chequear disco.

Función: Nos realiza un cheque de la unidad introducida como parámetro en el comando. Mostrándonos así el estado del disco o disquete.

Sintaxis: chkdsk [unidad:] [fichero]

Comando: DISKCOPY

Etimología: Deriva de disk (disco) y copy (copiar) = copia de discos.

Función: Copia el contenido total de un disco o disquete, sirve para hacer copias de seguridad, etc.

Sintaxis: diskcopy <unidad_origen:> <unidad_destino:>

Comando: DISKCOMP

Función: Tras realizar una copia de disquetes podemos realizar una verificación, para ver si ha copiado todos los contenidos, comparando. Este comando compara discos o disquetes.

Sintaxis: diskcomp <unidad1:> <unidad2:>

Comando: LABEL

Función: Es una orden que permite cambiar o borrar el nombre de la etiqueta que tiene asignado un disco o disquete cuando este es formateado. La etiqueta es simplemente un nombre asignado por el usuario para identificar el disco o disquete. Generalmente es bueno que dicho nombre haga alusión a su contenido.

Sintaxis: label [etiqueta de volumen]

Donde, poniendo solo label nos visualizaría el nombre de la etiqueta, actual. Y especificando ya ese parámetro nos lo podria directamente.

Comando: VOL

Función: Este comando nos muestra en pantalla, prácticamente la misma

información que el anterior, haciendo la salvedad de que este también nos muestra el numero de serie que se le es asignado al disco o disquete. Este numero de serie no es modificable puesto que la etiqueta si lo es y ha de haber alguna forma con la que siempre el sistema pueda identificar un disco. Esto lo hace refiriéndose a ese numero de serie (no modificable).

Sintaxis: vol

Comando: DEFRAG

Función: Defragmenta y obtimiza el disco, almacenando los ficheros en clusters secuenciales, lo que optimiza el rendimiento del sistema.

Sintaxis**: defrag**

Comando: SCANDISK

Función: Comprueba la integridad de los datos almacenados basándose en el estado del disco que almacena estos datos.

Sintaxis**: scandisk**

Comando: MSBACKUP

Función: Realiza copias de seguridad. Permite realizar una copia de seguridad de todos o parte de los archivos que se encuentran en el disco duro.

Sintaxis: MSBACKUP [nombre del fichero y especificaciones]

Comando: RESTORE

Función: Restablece los ficheros de los discos de seguridad. Es un comando complementario al anterior.

Sintaxis: RESTORE disk-1: disk-2: [archivos]

Comandos para gestiones varias

Comando: PROMPT

Sintaxis: **prompt [texto o parametros]**

Definición: Este comando sirve para mostrar en pantalla un texto deseado. Mediante una serie de signos podemos hacer que sean restituidos por valores ya almacenados como la hora, la fecha, etc. Los que sabéis programar en scriptting veréis que el uso de estos signos es muy similar al uso de identificadores.

Ejemplo: C:\>prompt El verdadero poder se encuentra en OrioN ScripT.$_ Hoy es dia $d y son las $._Telee una nueva orden pg .

Signos devolutorios de valores:

$ Este signo, es el que siempre se ha de colocar, una vez que es leído por el MSDOS, sabe que ha de sustituir la cadena de carácter que le sigue por un valor almacenado.

$_ Procesa el efecto de retorno de carro, o lo que es lo mismo un salto de línea (intro).

$b Procesa una canalización (|).

$d Muestra en pantalla la fecha del sistema.

$e Representa el código ascii 27, tecla escape.

$g Muestra en pantalla el carácter “>”.

$h Procesa un retroceso, la tecla back space. Elimina el carácter anterior.

$l Muestra en pantalla el carácter, “<”.

$n Visualización de la unidad de trabajo por defecto.

$p Visualiza el directorio de trabajo incluyendo la unidad.

$q Muestra el carácter “=”.

$t Muestreo de la hora del sistema.

$v Muestra la versión del MSDOS con la que estamos operando.

Comando: CLS

Sintaxis: cls

Función: Sus iniciales vienen de clean screen (limpiar pantalla). Su uso es muy simple, se introduce el comando se pulsa intro, y se procesa un limpiada de pantalla. Solo queda el directorio en el que nos encontrábamos, situado en la parte superior de la pantalla.

Comando: VER

Función: Este comando nos muestra en pantalla la versión que estamos usando del MSDOS.

Etimología: De versión en ingles.

Sintaxis: ver

Comando: DATE

Etimología: De fecha en ingles

Función: Nos muestra en pantalla la fecha almacenada por el sistema de forma diaria. En una línea posterior nos facilita la posibilidad de cambiar la configuración de la fecha del sistema.

Sintaxis: [dd-mm-aa]

Comando: TIME

Etimología: Supongo que sabréis de donde viene jejeje.

Función: Funciona de la misma forma que el comando anterior. La única diferencia es que el anterior muestra la fecha y este la hora, resulta bastante evidente, no ☺. También nos permite la tarea de cambiar la hora en una línea posterior al muestreo de la hora actual almacenada por el sistema.

Sintaxis: time

Atributos de los ficheros
Atributos

Permiten asociar a los archivos unas características especiales (oculto, de sistema, de solo lectura, y de archivo).

Estas características pueden estar activadas o desactivadas en los archivos a los que se refieren.

Con este comando podrás visualizar y modificar atributos de ficheros.

1. **De solo lectura (r):** Protege a un fichero de forma que solo puede ser leido y no modificado ni borrado.
2. **De archivo (a) :** Sirve para saber si un determinado fichero ha sido o no modificado.
3. **Oculto (h):** Sirve para esconder un fichero.
4. **De sistema (s) :** Sirve para identificar los archivos propios del sistema, los cuales sirven para cargar el SO (sistema operativo).

Comando: ATTRIB

Para visualizar:

Sintaxis: attrib /s

Para modificar atributos:

Sintaxis: attrib <fichero> <+/-><a/h/s/a/r>

Para listar con el dir ficheros con atributos:

Sintaxis: dir /a:h

Donde **dir** es el comando de listado de archivos, **/a** es la salida al comando attrib que mirara solo los archivos **:h** con el modificador h (oculto).

Redireccionamiento y filtros

Dispositivos:

CON: Bajo este nombre MSDOS identifica al teclado y al monitor (principales dispositivos de entrada y salida). Identifica con el mismo nombre al monitor y al teclado porque son dispositivos unidireccionales, lo que quiere decir que por ellos solo se puede realizar o la acción de entrada de datos o de salida, con lo cual no hay problemas a la hora de identificarlos con el mismo nombre. El nombre CON viene de “CONsola”.

LPTn: Este identificativo se refiere a los puertos pararelos. El MSDOS consigue identificar hasta 3 puertos pararelos: LPT1, LPT2, LPT3.

Estos puertos pararelos son bidireccionales. Y suelen estar conectados a ellos dispositivos como la impresora, generalmente este esta en el LPT1. A la impresora también se le designa PRN.

COMn: Se emplea para identificar los puertos de comunicaciones. Los llamados puertos series, se denominan series, porque envía bit a bit de forma sucesiva, uno detrás de otro. Son dispositivos bidireccionales a los que se suelen conectar el raton, el MODEM, etc. MSDOS puede reconocer hasta 4 puertos serie: COM1, COM2, COM3, COM4.

NUL: No identifica a ningún dispositivo en concreto. Pero se crea para representar a un dispositivo ficticio para así simular la trasferencia o recepción de información.

Haber, en MSDOS no se trabaja solo con el monitor y el teclado como dispositivos de salida y entrada. MSDOS permite trabajar con otros dispositivos de entrada y salida como puede ser la impresora, etc. Esto se realiza gracias a los redireccionamientos del DOS. Con los cuales si queremos en vez de ver el resultado de un DIR en pantalla podemos redireccionar el comando y sacarlo por impresora, todo el listado. También podremos grabar ese dir en un fichero, etc.

Los **redireccionamientos** usan unos signos especiales con los que se identifican.

Operadores de redireccionamiento de salida: Es el signo de (mayor que): “>”. Este signo dirigirá el comando introducido en el DOS hacia un dispositivo de entrada o slaida. Por ejemplo, si ponemos en el DOS: a:\>dir>director.txt. Esto hará que salgan todo el listado de dir en el archivo especificado. Podemos hacer lo mismo pero en vez de director.txt ponemos el identificativo de la impresora LPT1 o PRN, de esta forma saldrá impreso el resultado del DIR.

Operadores de redireccionamiento de entrada: Es el signo de (menor que): “<”. Indica al DOS que debe tomar los datos de entrada desde un fichero que se especifique (por supuesto debe existir dicho fichero) o de otro dispositivo, en lugar de tomarlos por teclado. MSDOS cuenta con 3 ordenes tomadas específicamente para este tipo de redireccionamientos, hablamos de: **SORT** (ordena los datos de entrada), **MORE** (visualiza el contenido de los datos de entrada fraccionando su presentación en pantalla completa), **FIND** (localiza una cadena de caracteres especificos dentro de los datos de entrada).

Operadores de redireccionamientos de adicción a un fichero: Se usa el signo doble mayor que: “>>”. La diferencia entre este y el anterior de redireccionamiento a un fichero, radica que este redirecciona el resultado a un fichero y si este fichero contiene ya algo pues lo situá al final. Mientras que el anterior en caso de existir el fichero y de contener algo, lo borraba para meter los nuevos datos surjidos del redireccionamiento.

Filtros

MSDOS además de permitir redireccionar las ordenes a dispositivos de entrada y salida, también permite direccionar las ordenes a otras ordenes. Para ello cuenta con los FILTROS, para identificarlos usa el signo | (alt gr + 1). Los filtros con los que cuenta son: MORE, SORT, FIND.

SORT: Este filtro ordena los datos de entrada. Por defecto los ordena según la primera letra de los datos de salida. Se refiere a la primera letra de cada fila de toda la columna.

Sintaxis: SORT [/R] [/+numero]

/R: Indica a MSDOS que debe invertir el orden de la ordenación, es decir que orden descendentemente, de mayor a menor.

/+numero: Indica a MSDOS que en lugar de ordenar por el primer carácter (la primera columna), lo haga por el carácter que ocupe la posición que se le especifique.

FIND: Este filtro localiza una cadena de caracteres dentro de un fichero, por tanto , es aconsejable que se emplee con un fichero de texto. Este filtro recibe como

entrada datos dispuestos en filas y devuelve solo aquellas filas que contienen la secuencia de caracteres o cadena que se ha especificado en la orden.

Sintaxis: FIND /V /C /N /I <cadena> [fichero]

/V: Muestra las líneas que no contienen la cadena.

/C: Muestra las líneas que contienen la cadena.

/I: Omite mayusculas y minúsculas.

/N: Muestra las líneas de texto con la cadena y los numeros de esas líneas

MORE: Permite obtener el resultado de una orden de forma paginada, si el resultado es mayor que la pantalla (mas de 25 lineas). Para así poder visualizarlo todo.

COMANDOS BÁSICOS DE LINUX

Los comandos son esencialmente los mismos que cualquier sistema UNIX. En la tablas 3.1 y 3.2 se tiene la lista de comandos mas frecuentes. En la tabla 3.3 se tiene una lista de equivalencias entre comandos Unix/Linux y comandos DOS.

Comando/Sintaxis	**Descripción**	**Ejemplos**
cat *fich1* [...*fichN*]	Concatena y muestra un archivos	cat /etc/passwd
	archivos	cat dict1 dict2 dict
cd [*dir*]	Cambia de directorio	cd /tmp
chmod *permisos fich*	Cambia los permisos de un archivo	chmod +x miscript
chown *usuario:grupo fich*	Cambia el dueño un archivo	chown nobody miscript
cp *fich1...fichN dir*	Copia archivos	cp foo foo.backup
diff [-*e*]*arch1 arch2*	Encuentra diferencia entre archivos	diff foo.c newfoo.c
du [-*sabr*] *fich*	Reporta el tamaño del directorio	du -s /home/
file *arch*	Muestra el tipo de un archivo	file arc_desconocido
find *dir test acción*	Encuentra archivos.	find . -name ``.bak" -print
grep [-*cilnv*] *expr archivos*	Busca patrones en archivos	grep mike /etc/passwd

head *-count fich*	Muestra el inicio de un archivo	head prog1.c
mkdir *dir*	Crea un directorio.	mkdir temp
mv *fich1 ...fichN dir*	Mueve un archivo(s) a un directorio	mv a.out prog1
mv *fich1 fich2*	Renombra un archivo.	mv .c prog_dir
less / more *fich(s)*	Visualiza página a página un archivo.	more muy_largo.c
	less acepta comandos vi.	less muy_largo.c
ln [*-s*] *fich acceso*	Crea un acceso directo a un archivo	ln -s /users/mike/.profile .
ls	Lista el contenido del directorio	ls -l /usr/bin
pwd	Muestra la ruta del directorio actual	pwd
rm *fich*	Borra un fichero.	rm foo.c
rm -r *dir*	Borra un todo un directorio	rm -rf prog_dir
rmdir *dir*	Borra un directorio vacío	rmdir prog_dir
tail *-count fich*	Muestra el final de un archivo	tail prog1.c
vi *fich*	Edita un archivo.	vi .profile

Comandos Linux/Unix de manipulación de archivos y directorios tbl_comm_archivos

Comando/Sintaxis	**Descripción**	**Ejemplos**
at [*-lr*] *hora* [*fecha*]	Ejecuta un comando mas tarde	at 6pm Friday miscript
cal [[*mes*] *año*]	Muestra un calendario del mes/año	cal 1 2025
date [*mmddhhmm*] [*+form*]	Muestra la hora y la fecha	date
echo *string*	Escribe mensaje en la salida estándar	echo ``Hola mundo''
finger *usuario*	Muestra información general sobre	finger nn@maquina.aca.com.co
	un usuario en la red	
id	Número id de un usuario	id usuario

kill [*-señal*] *PID*	Matar un proceso	kill 1234
man *comando*	Ayuda del comando especificado	man gcc
		man -k printer
passwd	Cambia la contraseña.	passwd
ps [*axiu*]	Muestra información sobre los procesos	ps -ux
	que se están ejecutando en el sistema	ps -ef
who / rwho	Muestra información de los usuarios	who
	conectados al sistema.	

Comandos Linux/Unix más frecuentes tbl_comm_frec

Linux	**DOS**	**Significado**
cat	type	Ver contenido de un archivo.
cd, chdir	cd, chdir	Cambio el directorio en curso.
chmod	attrib	Cambia los atributos.
clear	cls	Borra la pantalla.
ls	dir	Ver contenido de directorio.
mkdir	md, mkdir	Creación de subdirectorio.
more	more	Muestra un archivo pantalla por pantalla.
mv	move	Mover un archivo o directorio.
rmdir	rd, rmdir	Eliminación de subdirectorio.
rm -r	deltree	Eliminación de subdirectorio y todo su contenido.

Equivalencia de comandos Linux/Unix y DOS tbl_equiv_comandos

Unidad 3: HERRAMIENTAS DE CONFIGURACION ANALISIS Y OPERACIÓN DEL SISTEMA DE COMPUTO

INTRODUCCION

La computadora hoy en día se ha vuelto una herramienta indispensable en Muchas áreas, lo mismo puede servir para calcular la distancia de la estrella más lejana de nuestro sistema solar como para la diversión y esparcimiento de un niño que la utiliza para jugar, al igual que el ama de casa la puede utilizar para llevar una gran colección de recetas de cocina, y como cualquier
herramienta necesita cuidados y tratos especiales.

¿QUÉ ES EL MANTENIMIENTO PARA PCS?

Es el cuidado que se le da a la computadora para prevenir posibles fallas, se debe tener en cuenta la ubicación física del equipo ya sea en la oficina o en el hogar, así como los cuidados especiales cuando no se está usando el equipo. Hay dos tipos de mantenimiento, el preventivo
y el correctivo.

TIPOS DE MANTENIMIENTO PARA LA PC

- ***Mantenimiento preventivo para PCs***

El mantenimiento preventivo consiste en crear un ambiente favorable para el sistema y conservar limpias todas las partes que componen una computadora. El mayor número de fallas que presentan los equipos es por la acumulación de polvo en los componentes internos, ya queéste actúa como aislante térmico.

El calor generado por los componentes no puede dispersarse adecuadamente porque esatrapado en la capa de polvo.
Las partículas de grasa y aceite que pueda contener el aire del ambiente se ezclan con elpolvo, creando una espesa capa aislante que refleja el calor hacia los demás componentes, conlo cual se reduce la vida útil del sistema en general.

Por otro lado, el polvo contiene elementos conductores que pueden generar cortocircuitos entrelas trayectorias de los circuitos impresos y tarjetas de periféricos.
Si se quiere prolongar la vida útil del equipo y hacer que permanezca libre de reparaciones pormuchos años se debe de realizar la limpieza con frecuencia.

- ***MANTENIMIENTO CORRECTIVO PARA PCS***

Consiste en la reparación de alguno de los componentes de la computadora, puede ser una soldadura pequeña, el cambio total de una tarjeta (sonido, video, SIMMS de memoria, entre otras), o el cambio total de algún dispositivo periférico como el ratón, teclado, monitor, etc.

Resulta mucho más barato cambiar algún dispositivo que el tratar de repararlo pues muchas veces nos vemos limitados de tiempo y con sobre carga de trabajo, además de que se necesitan aparatos especiales para probar algunos dispositivos.

Asimismo, para realizar el mantenimiento debe considerarse lo siguiente:

- En el ámbito operativo, la reconfiguración de la computadora y los principales programas que utiliza.
- Revisión de los recursos del sistema, memoria, procesador y disco duro.
- Optimización de la velocidad de desempeño de la computadora.
- Revisión de la instalación eléctrica (sólo para especialistas).
- Un completo reporte del mantenimiento realizado a cada equipo.
- Observaciones que puedan mejorar el ambiente de funcionamiento.

CRITERIOS QUE SE DEBEN CONSIDERAR PARA EL MANTENIMIENTO *A LA PC*

-

La periodicidad que se recomienda para darle mantenimiento a la PC es de una vez por semestre, esto quiere decir que como mínimo debe dársele dos veces al año, pero eso dependerá de cada usuario, de la ubicación y uso de la Computadora así como de los cuidados adicionales que se le dan a la PC.

Por su parte, la ubicación física de la computadora en el hogar u oficina afectará o beneficiará ala PC, por lo que deben tenerse en cuenta varios factores:

- ***Hogar***

Es necesario mantener el equipo lejos de las ventanas, esto es para evitar que los rayos del sol dañen a la PC, así como para evitar que el polvo se acumule con mayor rapidez, también hay que tratar de ubicar a la PC en un mueble que se pueda limpiar con facilidad, si en la habitación donde se encuentra la PC hay alfombra se debe aspirar con frecuencia para evitar que se acumule el polvo.

También no es conveniente utilizar el monitor como “repisa”, esto quiere decir que no hay que poner nada sobre el monitor ya que genera una gran cantidad de calor y es necesario disiparlo, lo mismo para el chasis del CPU.

- ***Oficina***

Los mismos cuidados se deben tener en la oficina, aunque probablemente usted trabaje en una compañía constructora y lleve los registros de materiales, la contabilidad, los planos en Autocad, etc. Esto implicaría que la computadora se encuentre expuesta a una gran cantidad de polvo, vibraciones y probablemente descargas eléctricas, así mismo la oficina se mueve a cada instante, hoy puede estar en la Ciudad de México y en dos semanas en Monterrey, por lo mismo el mantenimiento preventivo será más frecuente.

Consideraciones finales:

- No exponer a la PC a los rayos del sol.
- No colocar a la PC en lugares húmedos.
- Mantener a la PC alejada de equipos electrónicos o bocinas que produzcan campos magnéticos ya que pueden dañar la información.
- Limpiar con frecuencia el mueble donde se encuentra la PC así como aspirar con frecuencia el área si es que hay alfombras.
- No fumar cerca de la PC.
- Evitar comer y beber cuando se esté usando la PC.
- Usar “No-Break” para regular la energía eléctrica y por si la energía se corta que haya tiempo de guardar la información.
- Cuando se deje de usar la PC, esperar a que se enfríe el monitor y ponerle una funda protectora, así como al teclado y al chasis del CPU.
- Revisión de la instalación eléctrica de la casa u oficina, pero esto lo debe de hacer un especialista.

MATERIAL, HERRAMIENTAS Y MESA DE TRABAJO

Como ya se había explicado anteriormente el mantenimiento preventivo ayudará a alargar el
buen funcionamiento de la PC, para ello se tiene que contar con una mesa de trabajo, la cual
preferentemente no debe de ser conductora (que no sea de metal o similar), se debe de tener el
área o mesa de trabajo libre de estorbos y polvo.
También es importante contar con las herramientas y material adecuado, todo esto para poder
facilitar el trabajo:

MEDIDAS DE SEGURIDAD

Estas medidas aunque le parezcan básicas son vitales para la seguridad de su equipo de cómputo y su seguridad personal:

- Antes de abrir cualquier computadora es necesario revisarla para poder detectar posibles fallas, por lo cual hay que encender la computadora y probar todas y cada una de las aplicaciones, revisar las unidades de disco flexible y la unidad de CD-ROM, así como verificar que cada una de las teclas del teclado funcionen adecuadamente, y que tanto el ratón como los botones se desplacen sin ningún problema.

- Si detectó algún problema tome nota e infórmele al dueño del equipo.

- Antes de quitar los tornillos es recomendable que desconecte la computadora de la energía, quite todos los cables exteriores, tomando nota del lugar de donde los quitó.

- Retire los tornillos e introdúzcalos en el bote para rollo fotográfico (así se evita perder los tornillos), asegúrese de utilizar el desarmador adecuado.

- Quite la tapa de la computadora.

- Si el CPU es mini-torre ‘‘acuéstelo’’ para poder trabajar con comodidad y seguridad.

- Antes de quitar cualquier componente observe con cuidado la parte interna de la PC,tome nota de la colocación de las tarjetas, para que cuando termine el mantenimiento preventivo las coloque en el lugar exacto de donde las sacó.

- Ya que haya tomado nota de todos los pequeños detalles proceda a colocarse la pulsera antiestática, esto es para evitar dañar alguna tarjeta.

- Quite el tornillo que sujeta a la tarjeta con el chasis de la PC e introdúzcalo también en el botecito, tal vez el tornillo sea un poco más pequeño que los tornillos del chasis, si es así colóquelo en otro botecito, etiquete los botecitos con cinta adhesiva para mayor control.
- Cuando saque alguna tarjeta y ya la haya limpiado colóquela dentro de una bolsa antiestática, lo mismo para todas las tarjetas.

NOTA. Es recomendable que no quite el disco duro, microprocesador y fuente de poder.

Reglas básicas de configuración e instalación física de dispositivos

Ya que haya limpiado todas las tarjetas, incluyendo la tarjeta principal, el siguiente paso es volver a armar la PC, para lo cual se le recomienda lo siguiente:

- Nunca introduzca una tarjeta en una ranura que no le corresponde, por ejemplo; una tarjeta ISA nunca entrará en una ranura PCI pero si usted se empeña en meterla puede dañar la tarjeta o la ranura.

- El mismo procedimiento se lleva a cabo para los conectores de alimentación, en tanto tienen una forma especial o particular que impide introducirlos al revés, observe muy bien el dispositivo que necesita alimentación y verá que tiene la misma forma que el conector (sólo que a la inversa), es decir, si el conector del dispositivo es hembra
forzosamente necesita insertarle un conector macho y así sucesivamente.

- Si desconectó los conectores P8 y P9 de la fuente de alimentación de la tarjeta principal, siga esta sencilla recomendación: los cables negros tienen que ir juntos, no los invierta ya que pueden dañar el equipo.

- Cuando inserte los cables tipo Listón tiene que seguir la "Ley del Pin 1", esta ley o regla implica la manera como se tiene que colocar el cable o Bus, observe con cuidado sus cables tipo Listón y podrá ver que en uno de los extremos el cable tiene un filamento rojo, ese filamento indica que es el Pin 1, ahora en su dispositivo (disco duro, unidad de disco flexible o CD-ROM) en la parte exterior cerca del lugar donde se inserta el cable tiene que ver un número 1 o una especie de flecha, esa señalización indica que es el Pin 1; en pocas palabras tiene que coincidir el filamento rojo con el No. 1 o la flecha indicada en el dispositivo.

- Colocar las tarjetas en el lugar exacto de donde las sacó, así evita alterar la configuración que ya se tenía antes.

- Antes de cerrar el equipo verificar que funcione adecuadamente.

Recomendaciones:

- Nunca introducir nada a la fuerza, ya que se pueden dañar los conectores y los dispositivos, sólo entra de una manera.

- Colocar todo como estaba antes de desarmar la PC.

- Seguir la regla del Pin 1.

MANTENIMIENTO PREVENTIVO AL CPU

El medio ambiente que rodea a la computadora personal encuentra en ella un imán de polvo,se preguntará y esto en qué me afecta. Pues resulta que el polvo, aunado a un ambiente húmedo o muy seco puede ser un magnífico conductor eléctrico, lo cual puede provocarpequeñas fallas en los componentes electrónicos de la computadora personal, asimismo el polvo acumulado reduce la eficiencia de los ventiladores de enfriamiento y puede actuar como un manto aislante que conserva el calor y no permite que la irradiación de éste se aleje de los componentes.

De este modo, se debe limpiar el sistema de acuerdo con una calendarización que ya se comentó en capítulos anteriores, tomando en cuenta que dependiendo del medio ambiente que rodee la computadora dependerá la periodicidad con que se lleve a cabo esta tarea.

Ahora bien, si ya se está dispuesto a dar mantenimiento a la computadora, será conveniente establecer medidas de seguridad y más o menos determinar cuál será el área de trabajo ideal para abrir la computadora. La mayor de las veces que uno realiza un trabajo, cualquiera que sea éste, es necesario siempre contar con todo el material, herramientas y área de trabajo adecuados para llevar a buen Término dicha tarea. Un ejemplo muy simple es el siguiente: si al retirar una tuerca para remover una pieza mecánica, no cuento con una llave adecuada, y por falta de tiempo utilizo unas pinzas de presión, de momento se soluciona el problema, pero al no utilizar la llave adecuada se pueden ocasionar problemas que van desde el maltrato de la tuerca en el menor de los casos, y en el peor su deformación por la aplicación excesiva de presión, con la consecuencia de quedar inutilizada y tener que retardar el término de la tarea.

El ejemplo anterior muestra de una manera muy simple el problema que se puede ocasionar sino no se cuenta con la herramienta adecuada. En el caso de equipo de cómputo el usoinadecuado de herramientas puede causar conflictos muy sencillos como cambiar un tornillo, y tan graves como cambiar una tarjeta electrónica (Madre, video, sonido, etcétera).

La mesa de trabajo es una parte importante para poder realizar eficientemente el trabajo de limpieza así como su amplitud es una característica importante, ya que es necesario contar con el espacio adecuado para no correr el riesgo de que se caigan los componentes retirados del gabinete (cables, tarjetas de expansión, etcétera).

Una iluminación adecuada es indispensable para poder observar las áreas que se limpiarán, a la par de una mejor identificación de los componentes de la computadora para evitar confusiones al momento de conectar los diferentes cables que hay dentro del sistema.

En el mercado hay diferentes tipos de destornilladores, debido al diseño de la punta que tienen:
plano, de cruz, estrella y de caja.

De todos los tipos de destornilladores mencionados se necesitarán, por lo menos un juego de tres medidas en cada uno de los casos, en cuanto a los destornilladores de caja si conviene tener un juego completo.

Las pinzas son una herramienta sumamente útil ya que ayudan a llegar a esos rincones donde a veces no entran sus dedos y es necesario tomar o conectar algo de ahí. Tambien sirven para enderezar los contactos que a veces por las prisas doblamos.

Hay varios tipos de pinzas, de las cuales ocupará sólo las de punta y corte, ambas por lo menosen dos tamaños, pequeñas y medianas.Muchos de los circuitos del interior de la computadora son susceptibles de sufrir daños a causa de la electricidad estática. Una simple descarga puede inutilizar los circuitos integrados, lo cual a su vez puede repercutir en un mal y hasta inhabilitar el equipo. Debido a que la electricidad estática puede inclusive generarse en el cuerpo humano —esto variará dependiendo de cada uno como individuo— se necesitan tomar unas cuantas precauciones cuando se estén manejando componentes de la computadora, y una de ellas es ocupar la pulsera antiestática.

La pulsera antiestática es un dispositivo que se adapta a su muñeca y lo conecta a una fuente de tierra (como la parte metálica de una caja) para mantenerlo libre de electricidad estática. Si tiene alfombra en el cuarto donde está trabajando con la computadora, tome sus precauciones contra la descarga de electricidad estática que definitivamente se generará en su cuerpo. En cualquier caso, no arrastre demasiado los pies mientras se encuentre trabajando con la computadora. Se generará menos electricidad estática de esta manera.Una vez que se han tomado las anteriores recomendaciones, hay que comenzar a darle mantenimiento al CPU y sus componentes. **No hay que olvidar apagar la computadora y desconectar el cable de alimentación de la toma de energía.**

Tarjeta Madre

Las mejores herramientas para esta labor son una brocha de cerdas rígidas limpia, una aspiradora y un producto limpiador-desengrasante. Utilice la brocha para remover el polvo adherido a los componentes para que la aspiradora pueda a su vez quitarlo. Aunque se debe de aspirar todo el polvo que se encuentre dentro del sistema hasta donde sea posible (sin exagerar al remover puentes, disipadores adheridos por pegamento o grapas, etc.), hay que poner especial énfasis en las siguientes áreas:

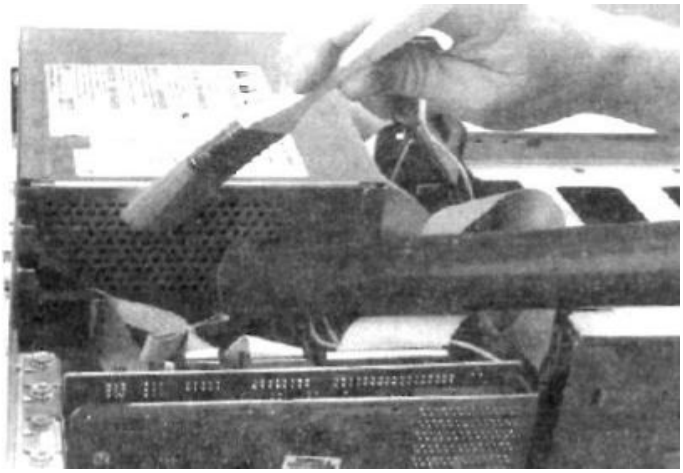
- Ventilador del CPU. Éste puede acumular casi tanto polvo como la fuente de poder, y como el CPU genera demasiado calor, es importante conservar limpio el ventilador para mantener en buen estado su capacidad de enfriamiento.

Por lo tanto, si a simple vista se nota que éste ha sufrido deterioro por el paso del tiempo, o usted a notado que produce un ruido excesivo, será necesario que lo cambie, ya que el calentamiento excesivo en el CPU puede provocar fallos del sistema.

- Ranuras de expansión (ISA, PCI y AGP). Al mantener el polvo fuera de estas ranuras se asegura una buena calidad de conexión, si se instala posteriormente una tarjeta adaptadora en la ranura.

Una vez retirado el polvo excesivo se puede aplicar un producto que acabe de retirar la suciedad de la tarjeta y que normalmente contiene una sustancia desengrasante; esto sirve para evitar que pequeños residuos de grasa provoquen la acumulación temprana de polvo.

PRECAUCIÓN. Se deberá resistir la tentación de invertir el flujo del aire de la aspiradora o emplear aire comprimido para soplar el polvo fuera de la computadora. En primer lugar, sólo se lograría soplar el polvo de regreso a la habitación, de manera que puede caer otra vez dentro de la computadora. Sin embargo es más importante el hecho de que el polvo tiene la tendencia a abrirse paso dentro de las unidades lectoras de disco flexible, ranuras de expansión y otros lugares difíciles de alcanzar. Además, cuide que la brocha y la boquilla de la aspiradora no golpeen ni dañen algo.



Limpiando la fuente de poder

. **SIMMs y DIMMs de memoria RAM**

Para poder limpiar los SIMMs y DIMMs es necesario desmontarlos de la Tarjeta madre, a continuación se explica cómo hacerlo.

Extraer un SIMM no es una tarea muy difícil, para extraerlos de la ranura, basta con presionar las lengüetas laterales. Si no es posible hacerlo con los dedos, puede hacerse con la ayuda de un destornillador plano, teniendo mucho cuidado de no dañar ningún componente.

En especial hay que evitar clavar el destornillador o rayar con él la superficie de la tarjeta madre

El procedimiento para retirar el polvo de estos dispositivos es exactamente igual al estudiado con anterioridad (Tarjeta Madre), sólo habrá que añadir que en caso de que las terminales se encuentren sucias se recomienda limpiarlas con una goma de lápiz, asegurándose de que no sea demasiado dura para no maltratar las terminales. Acto seguido se podrá aplicar sobre los mismos el producto desengrasante para eliminar cualquier residuo de grasa que pudiera existir.

Se debe tener cuidado de tomar por los bordes los SIMMs y DIMMs para evitar posibles daños por descarga de electricidad estática generada por nuestro cuerpo. Es importante recalcar lo anterior ya que a veces estos dispositivos no se dañan de inmediato, pero se van degradando poco a poco, reduciendo así la vida útil de éstos.
Una vez acabado el proceso de limpieza, hay que volver a colocar los SIMMs, lo cual implica un proceso donde habrá que observar que éstos tienen una pequeña muesca en uno de los lados y en la base de la ranura donde se inserta, hay una pequeña rebaba de plástico que permite insertar el modulo de la memoria únicamente cuando coincide con esta rebaba.

Si esta operación se realiza correctamente, se empuja el módulo de memoria hasta que las lengüetas hacen un pequeño chasquido cuando se sitúan en su posición y aseguran el módulo de memoria.

Unidades lectoras y de almacenamiento

Disco duro

Por lo regular, no hay nada que hacer para limpiar un disco duro, de hecho, si se llegara a abrir un disco duro, en ese momento se haría inmediatamente inservible, ya que la mínima partícula de polvo o del medio ambiente, pueden destruir la cabeza de un disco duro. Por tanto, la limpieza del disco duro, solamente implica retirar el polvo depositado sobre la superficie externa con una brocha y aspiradora.

Unidad lectora de disco flexible

Otro dispositivo que se debe de limpiar cada cierto tiempo es la unidad lectora de disco flexible de la computadora. A diferencia de las cabezas de un disco duro, que se desplazan sobre el disco en un cojín de aire, las de una unidad lectora de disco flexible descansan sobre la superficie del medio magnético del disco flexible. De este modo, la cabeza tiene la tendencia a acumular en forma progresiva la suciedad del disco. Si las cabezas llegan a ensuciarse en demasía, la unidad no podrá leer ni escribir en el disco.

La limpieza de la unidad lectora no requiere que se desarme nada. En vez de ello, requiere de un limpiador especial, que se puede adquirir en cualquier tienda de productos de computación.

El disco limpiador tiene el aspecto de un disco normal, sólo que la parte interior de la cubierta del disco está hecha de una tela suave y porosa en lugar del substrato plástico/magnético empleado en un disco normal. El conjunto de limpieza incluye un líquido que se aplica en la tela del disco. Posteriormente se introduce este disco en la unidad lectora y se intentará tener acceso a él, mediante el comando **DIR A:** si está en ambiente de DOS, o presionar dos veces el botón izquierdo del ratón en la unidad A: de la ventana de Mi PC, en Windows 95, 98 y Windows NT 4.0.

Fuente de alimentación

Nunca abra la fuente de poder para tratar de limpiar el interior, aunque se puede y debe aspirar el polvo de los orificios laterales de la fuente. Esto ayuda al buen funcionamiento del ventilador de la misma y lo capacita para sacar más aire del gabinete. Además en la parte posterior de la fuente de poder, se puede aspirar el polvo acumulado sobre la superficie de las aspas del ventilador. Tal vez sea posible retirar temporalmente la protección de alambre que lo cubre (si es movible), para poder tener acceso a las aspas y remover el polvo con la brocha de cerdas firmes y finalizar con la aspiradora, pero asegúrese de volver a colocar la protección cuando haya acabado la limpieza.

Tarjetas en el sistema

Para poder realizar la limpieza de estos dispositivos será necesario desmontarlos de las ranuras de expansión, lo cual sólo implica retirar un tornillo que fija la tarjeta a la estructura del gabinete y evita que se desprenda.

El procedimiento para retirar el polvo de estos dispositivos es exactamente igual al estudiado con anterioridad (Tarjeta Madre), sólo debe añadirse que en caso de que las terminales se encuentren sucias se recomienda limpiarlas con una goma de lápiz, asegurándose de que no sea demasiado dura para no maltratar las terminales. Acto seguido se podrá aplicar sobre los mismos el producto desengrasante para eliminar cualquier residuo de grasa que pudiera existir.

Se debe tener cuidado de tomar por los bordes laterales las tarjetas para evitar posibles daños por descarga de electricidad estática generada por nuestro cuerpo. Es importante recalcar lo anterior ya que a veces estos dispositivos no se dañan de inmediato, pero se van degradando poco a poco, reduciendo así la vida útil de éstos.

El proceso de montaje de las tarjetas, al igual que el desmontaje no representa mayor problema más que introducir la tarjeta a su ranura, la mayor dificultad consistiría en que entrara muy ajustada, pero incorporando primero una de las esquinas y después el resto de la tarjeta en la ranura se soluciona el problema. **Asegúrese de que inserta la tarjeta en la ranura adecuada.**

PARTES ELEMENTALES DE UNA COMPUTADORA

Una computadora está compuesta por partes mecánicas y electrónicas, las cuales en conjunto la hacen funcionar, cada parte de la computadora recibe un nombre específico de acuerdo con la función que desempeña.

CPU (Unidad Central de Proceso)

El CPU es un microprocesador o chip que se coloca en la Tarjeta Madre, el CPU se encarga de procesar la información y para ello cuenta con dos sub-unidades: Unidad de Control y Unidad Aritmética Lógica.

- ***Unidad de Control***

Analiza y ejecuta cada instrucción del programa, controla las actividades de los periféricos, tales como un disco o una pantalla de presentación. A partir de señales que recibe del CPU, ejecuta las transferencias físicas de datos entre la memoria y el dispositivo periférico, se encarga de controlar todo el flujo de información.

- ***Unidad Aritmética Lógica (UAL)***

Circuito de alta velocidad que realiza las comparaciones y los cálculos. Los números son transferidos desde la memoria a la UAL (Unidad Aritmética Lógica) para realizar los cálculos, cuyos resultados son retransferidos a la memoria, los datos alfanuméricos son enviados desde la memoria a la UAL para su omparación, es la encargada de realizar todas las operaciones tanto aritméticas como lógicas.

Microprocesador Pentium II.

BIOS (Basic Imput/Output System)

Es un sistema básico de entrada y salida. Es un conjunto de rutinas de software (programa), que contienen las instrucciones detalladas para activar los dispositivos periféricos conectados a la computadora. La rutina de “autoarranque” del BIOS es responsable de probar la memoria en el arranque y de la preparación de la computadora para su operación.

CMOS (Complementary Metal Oxide Semiconductor)

Es un circuito integrado para procesadores y memorias, como utiliza poca batería es ideal para mantener al BIOS.

Tarjeta Madre (Mother Board o Tarjeta Principal)

Es la tarjeta principal o base, es un circuito impreso con dispositivos electrónicos que contiene ranuras de expansión que aceptan otras tarjetas adicionales.

La tarjeta principal contiene los conectores (zócalos) del CPU y el co-procesador matemático,cabe mencionar que el co-procesador matemático se encuentra en las 486SX y menores; los conectores de la memoria, el controlador del teclado, los chips de soporte, los puertos en serie o paralelo, las unidades de ratón y de disco pueden o no encontrarse presentes en la tarjeta principal, si no están son controladores independientes que se colocan en una ranura de expansión, es decir es una tarjeta controladora de puertos.

Diferentes clases de Tarjeta Madre

Una primera distinción la tenemos en el formato de la placa, es decir, en sus propiedades físicas.
Dicho parámetro está directamente relacionado con la caja, o sea, la carcasa del ordenador.
Hay dos grandes estándares: ATX y AT.

La segunda distinción la haremos por el zócalo del CPU, así como los tipos de procesador de soporte y la cantidad de los mismos. Tenemos el estándar tipo 4 o 5 para Pentium, el tipo 7 para Pentium y MMX, el Super 7 para los nuevos procesadores con BUS a 100 Mhz, el tipo 8 para Pentium Pro, el Slot 1 para el Pentium II, el Celeron, y el Slot 2 para los Xeon. Éstos son losmás conocidos.

La siguiente distinción la haremos a partir del CHIPSET que utilicen. El CHIPSET es un conjunto de circuitos integrados diseñados para trabajar junto con el microprocesador, con el fin de ejecutar una determinada función. Los más populares son los de Intel. Éstos están directamente relacionados con los procesadores que soportan; en este caso, para el Pentium están losmodelos FX, HX, VX y TX.

Para Pentium PRO los GX, KX y FX. Para Pentium II y sus derivados, además del FX, los LX, BX, EX, GX y NX. Para Pentium MMX se recomienda el TX, aunque es soportado por los del Pentium 'Classic'.

También existen placas que usan como CHIPSET el de otros fabricantes como VIA, SIS, UMC o Ali (Acer).
El siguiente parámetro es el tipo de BUS. Hoy en día el auténtico protagonista es el estándar PCI de 32 bits en su revisión 2.1, pero también es importante contar con alguna ranura ISA de 16 bits, pues algunos dispositivos como modems internos y tarjetas de sonido todavía no se han adaptado a este estándar, debido básicamente a que no aprovechan las posibilidades de ancho de banda del mismo.

También existe un PCI de 64 bits, aunque de momento no está muy visto en el mundo PC.Otros tipos de bus son el ISA de 8 bits, no usado ya, por ser compatible con el de 16 bits, el EISA, usado en algunas máquinas servidoras sobre todo de Compaq, el VL-Bus, de moda en casi todos los 486, o el MCA, el famoso bus microcanal en sus versiones de 16 y 32 bits patrocinado por IBM en sus modelos PS/2.

Otra característica importante es el formato y cantidad de zócalos de memoria que admite. En parte viene determinado por el chipset que utiliza. La más recomendable es la DIMM en formato SDRAM (Synchronous Dynamic RAM) con 3 zócalos. En el caso de módulos SIMM de 72 contactos el formato EDO RAM (Extended Data Output RAM) con 4 zócalos manejado por pares.

ATX (Advanced Technology Extended)

El estándar ATX es el más moderno y el que mayores ventajas ofrece. Está promovido por Intel,unque es una especificación abierta, que puede ser usada por cualquier fabricante sin necesidad de pagar regalías. La versión utilizada actualmente es la 2.01.

Entre las ventajas de la placa cabe mencionar una mejor disposición de sus componentes, la cual se obtiene básicamente girándola 90 grados. Permite que la colocación de la CPU no moleste a las tarjetas de expansión, por largas que sean. Otra ventaja es que se encuentra un solo conector de alimentación, que además no se puede montar al revés. La memoria está colocada en un lugar más accesible.

El CPU está colocado al lado de la FA (Fuente de Alimentación) para recibir aire fresco de su ventilador.
Los conectores para los dispositivos IDE y disqueteras quedan más cerca, reduciendo la longitud de los cables, además de estorbar menos la circulación del aire en el interior de la caja.

Aparte de todas estas ventajas, dicho estándar da la posibilidad de integrar en la placa base dispositivos como la tarjeta de video o la tarjeta de sonido, pero sacando los conectores directamente de la placa, para que proporcione un diseño más compacto, y sin necesidad de perder ranuras de expansión.

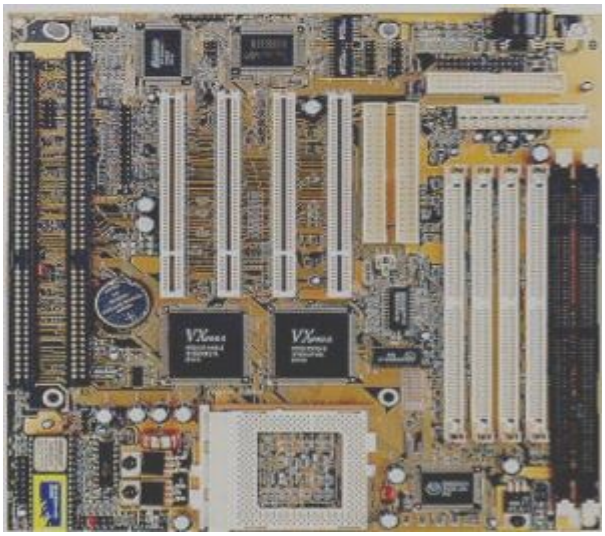
Así podemos tener integrados los conectores para teclado y ratón tipo PS/2, serie, paralelo o USB que son habituales en estas placas, pero también para VGA, altavoces, micrófono, etc., sacrificando apenas un poco de espacio.

AT (Advanced Technology)

Este formato está basado en el original del IBM PC-AT, pero de dimensiones más reducidas gracias a la mayor integración en los componentes de hoy en día, aunque físicamente compatible con aquél.

A la fecha sigue siendo el más extendido. En este tipo de placas es habitual el conector “DIN”. Para teclado. Entre sus ventajas cabe destacar el mejor precio tanto de éstas como de las cajas que las soportan, aunque esta ventaja desaparecerá en la medida que se vaya popularizando sucontrincante.

PARTES DE UNA TARJETA MADRE AT



1. Ranuras de expansión o slots PCI.
2. Puertos o COMs para ratón (mouse) y/o Módem (Modulador Demodulador).
3. Conector para teclado.
4. Conectores P8 y P9.
5. Ranuras de expansión o slots ISA.
6. Zócalos o bancos de memoria para SIMMs.
7. Conectores IDE para discos duros o CDs.
8. Zócalos o bancos de memoria para DIMMs.

9. Zócalo del microprocesador.
10. Conector de discos flexibles.
11. BIOS o sistema básico de entrada y salida.
12. Chipset.
13. Pila que alimenta al BIOS.

SIMMs y DIMMs de memoria RAM

- ***SIMMs (Single In Line Module Memory)***

Son unas pequeñas tarjetas con un conjunto de chips, que aumentan la memoria RAM de la computadora, hay diferentes capacidades y velocidades, las capacidades pueden ser de 1Mb, 4Mb, 16Mb, 32Mb y 64Mb. Los tiempos de acceso pueden ser de 80, 70, 60 o incluso 50 ns (nano segundos).

- ***DIMMs (Dynamic In Line Module Memory)***

También son chips de memoria, sólo que son un poco más largos que los SIMM, las capacidades son un poco mayores, los tiempos de acceso disminuyen en comparación de los SIMM, las capacidades pueden ser desde 16Mb, 32Mb, 64Mb y 128Mb. Los tiempos de acceso pueden ser de 60 y 10ns (nano segundos).

Interfaz de dispositivos

Todos los dispositivos necesitan interaccionar con el resto de la computadora, es decir necesitan cambiar información entre sí, pero eso sería imposible de no existir un medio de comunicación o una interfaz que implica un “camino” por el cual se pueden comunicar los dispositivos, básicamente se cuentan con dos tipos de interfaz de dispositivos:

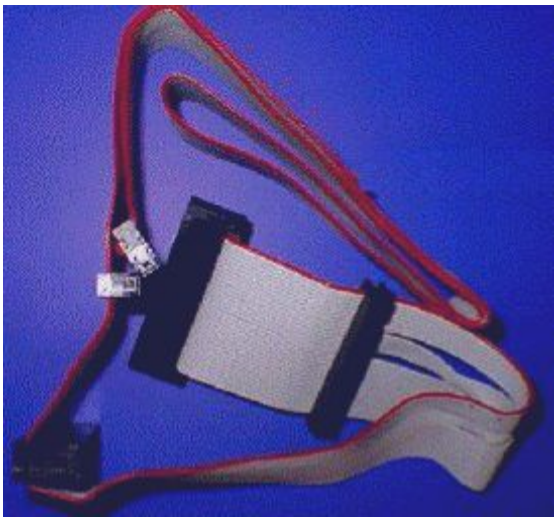
- ***Cable tipo listón o Bus para discos duros y unidades de disco compacto***

El cable para los discos duros y los CD-ROM es el mismo, siempre y cuando los dos dispositivos sean IDE, como podrá verse en uno de los extremos del cable tiene un filamento rojo, eso indica que es el Pin 1, también los dispositivos cuentan con una señal o indicador que determina cómo se tiene que colocar el cable, por ejemplo el disco duro puede tener su señal en la parte externa del disco, observe con cuidado y podrá ver un número 1 o una especie de “flechita”, cuando usted conecte su cable con el dispositivo asegúrese de que e l filamento rojo esté colocado del lado del indicador o señal.

Cable tipo listón para disco duro o CD-ROM.

- **Cable tipo listón o Bus para unidades de disco flexible**

Las unidades de disco flexible también necesitan una interfaz, sólo que ésta es un poco diferente a la de los discos duros o unidades del disco compacto, usted puede conectar dos unidades de disco flexible iguales, es decir puede tener dos unidades de 3½" o dos unidades de 5¼", una característica especial de estos cables es que tiene una pequeña "torcedura", después de localizar esa torcedura se considera que ahí se inserta el dispositivo como unidad A, también puede ver que tiene un filamento rojo. Siga el mismo procedimiento cuando los conecte a sus unidades.



Cable tipo listón para unidades de disco flexible.

SISTEMA DE ALMACENAMIENTO

Las unidades de almacenamiento están asociadas con una pieza de hardware a la que se denomina disco. Los tipos más comunes de discos son el disco duro (hard disk), disquete y disco compacto.

Disco duro (hard disk)

Es un disco magnético hecho de metal y cubierto con una superficie de grabación magnética, los discos duros pueden ser fijos o removibles, son unidades de almacenamiento; debido a que el tipo de almacenamiento es magnético se debe tener mucho cuidado al manejarlo o de tenerlo cerca de algún dispositivo que genere campos magnéticos (bocinas, imanes, etc.) pues éstos pueden borrar la información almacenada en el disco. Existen dos tipos de estándares en la interfaz de discos duros, los IDE y los SCSI.

- ***IDE (Integrated Drive Electronics)***

Cronológicamente, primero se encuentran los discos IDE con su limitación a 528 Mb. Sólo es posible conectar hasta dos de ellos.

Después vinieron los discos EIDE (Fast ATA), desarrollados por la compañía Western Digital,compatibles con los primeros, pero con algunas mejoras, basadas en la especificación ATA-2,que ya soporta unidades de CD-ROM (ATAPI) y de cinta.Otra mejora importante es el soporte de dos canales (cables tipo listón o Bus) para conectarhasta cuatro unidades. Además se definen varios modos de transferencia de datos, que lleganhasta los 16,6 Mb./seg. como el PIO-4, o mejor aun el *DMA-2, que soporta la misma tasa perosin intervención del CPU.

La última especificación, desarrollada por Quantum, es la Ultra *DMA/33 UltraATA), que permite transferencias DMA* a 33 Mb./seg.

***DMA.-** Es un conjunto de circuitos especializados o microprocesador dedicado que transfiere datos de memoria a memoria sin utilizar el procesador principal, sin embargo el DMA puede utilizar periódicamente ciclos del microprocesador, los datos son transferidos mucho más rápido que utilizando el procesador para la transferencia de cada byte.

- **SCSI (Small Computer System Interface)**

En el caso de los discos SCSI, tenemos el primero, llamado SCSI-1, con un ancho de bus de 8 bits, aunque ya en esta primera especificación se incluían características muy destacadas, como la posibilidad de conectar hasta 7 dispositivos de todo tipo: discos, cintas, escáner, CD-ROM,etcétera.

Después viene el SCSI-2, que ya dispone de un ancho de Bus de 16 bits. El siguiente paso es el
Fast-SCSI, considerado el doble de espacio. Después viene el Wide SCSI, ya con un ancho de Bus de hasta 32 bits.

Disco duro.

Unidad lectora de discos flexibles (floppy disk)

Estas unidades pueden ser de 3½" y de 5¼" (ésta última se encuentra en desuso), son partes electrónicas y mecánicas y también están expuestas al polvo u otros factores externos que pueden dañar a estos componentes, por consiguiente también necesitan de un mantenimiento preventivo o correctivo para su buen funcionamiento.

Unidad de disco compacto (CD-ROM)

La unidad de CD-ROM ha dejado de ser un accesorio opcional para convertirse en parte integrante de la PC, sin la cual no se podría ni siquiera instalar la mayor parte del software que actualmente existe, por no hablar ya de todos los programas multimedia y juegos.

Existen lectores, grabadores y regrabadores. Los más flexibles son los últimos, ya que permiten trabajar en cualquiera de los tres modos, pero la velocidad de lectura, que es uno de los parámetros más importantes se resiente mucho, al igual que en los grabadores.

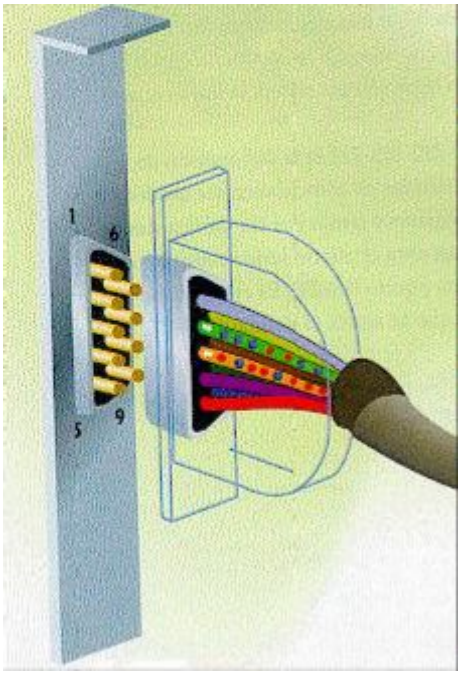
CD-ROM.

Puertos de comunicación

Son conectores externos que se utilizan para conectar un módem, mouse, impresora u otro dispositivo periférico, hay dos tipos de puerto: puerto serie y puerto paralelo.

Puerto serie

Utiliza una línea para enviar información, otra para recibirla; por su parte, existen líneas que regulan la información enviada por las otras dos líneas. El puerto serie se utiliza principalmente en la conexión del mouse o del módem. Envía información de un bit a la vez, esta transferencia es lenta pero suficiente para un mouse ya que la información que transmite un mouse es tan pequeña que la velocidad no es importante; para un módem es perfecta pues las líneas telefónicas no pueden transportar más que una señal a la vez. Los puertos seriales dentro de la PC pueden ser conocidos como puertos COM y utilizan conectores del tipo DB9.

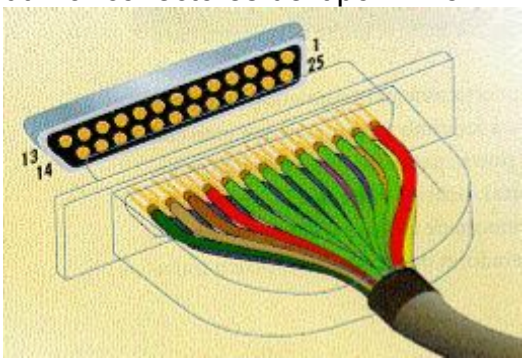


Puerto Serie.

Puerto paralelo

También es conocido como puerto Centronics, envía información a través de 8 cables paralelos simultáneamente en una sola dirección, envía varios bits de información, es mucho más rápido que el puerto serie. Este tipo de puerto es utilizado para impresoras.

Los puertos paralelos dentro de la PC pueden ser conocidos como LPT1 o LPT2 y utilizanconectores del tipo DB25.



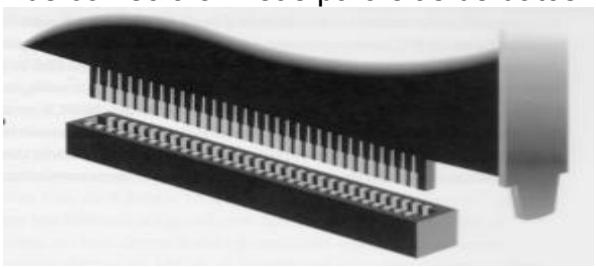
Puerto Paralelo.

Interfaz de datos (ranuras de expansión o Slots)

Es aquélla donde se insertan las tarjetas de dispositivos periféricos (tarjeta de video, tarjeta de sonido, tarjeta de red, fax/módem, etc.) para entrada o salida de información, estas unidades reciben el nombre de ranuras de expansión o Slots, y se encuentran sobre un circuito impreso llamado Bus o interfaz de datos, utilizado para transportar información a través de ellos.

- **Ranuras o Slots con Bus de 8 bits**

La información es transmitida a los Slots de expansión y otros componentes en el Bus con sólo 8 líneas paralelas de datos.



Ranura de 8 bits.

- ***Ranuras o Slots con Bus ISA (Industry Standard Architecture)***

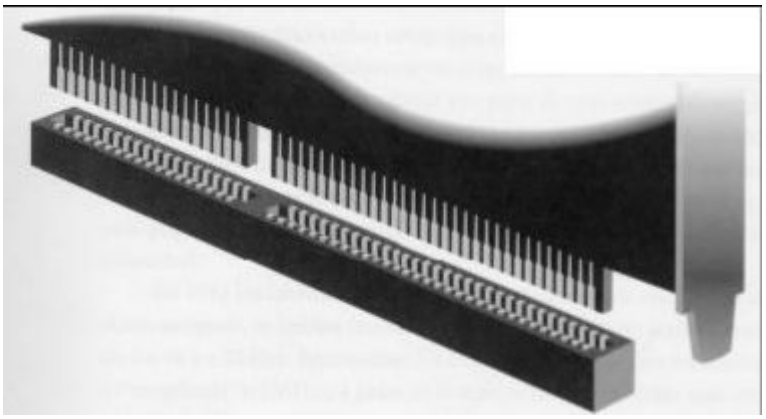
La información es también transmitida a lo largo de 8 o 16 líneas de datos, dependiendo de qué
clase de tarjeta sea utilizada en la ranura de expansión.

Ranura ISA.

- *Ranuras o Slots con Bus MCA (Microchanel Architecture)*

La tarjeta Microcanal de IBM utiliza 32 de sus 93 líneas para enviar y recibir datos. También incluye una circuitería que , al igual que Plug and Play*, facilita la instalación de la tarjeta.

***Plug and Play**: Windows 95 apoya la tecnología Conectar y Usar, que consiste en una norma que le permite instalar ciertos dispositivos de hardware rápido y fácilmente, por lo general sin necesidad de instalar controladores de dispositivo o de reconfigurar el sistema. Cuando compra dispositivos de hardware para la computadora, busque la identificación de Conectar y Usar (Plug and Play); en general esto hace que la instalación sea más fácil.



Ranura MCA.

- ***Ranuras o Slots con Bus EISA (Extended Industry Standard Arhitecture)***

Puede utilizar tarjetas de expansión diseñadas específicamente para trabajar con los 97 conectores de ranura divididos en dos niveles. Esta ranura transmite 32 bits de datos a la vez, las ranuras EISA admiten tarjetas de 8 y 16 bits.

Ranura EISA.

- ***Ranuras o Slots con Bus VESA de 32 bits (Video Electronic Standard Asociation)***

Se dividen en un conjunto de conectores basados en la ranura ISA y en un conjunto independiente de 32 pares adicionales de conectores más pequeños, que transportan la información del Bus local, trabajan con datos de 32 bits.

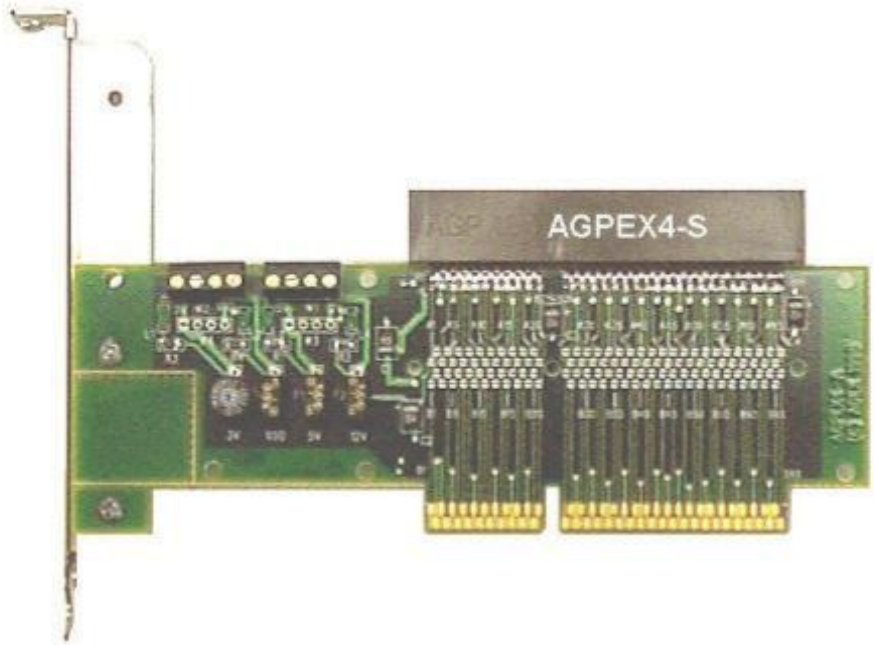
Ranura VESA.

- ***Ranuras o Slots con Bus PCI (Peripheral Component Interconection)***

Transmite información con un rango de 32 bits simultáneamente, estas ranuras de expansión no
aceptan de 8 o 16 bits ni tarjetas MCA.

- ***Ranuras o Slos con Bus AGP (Accelerated Graphics Port)***

Las ranuras AGP proveen un puerto de alta velocidad que sólo transporta información gráfica e,incorpora otras funciones para el manejo de gráficas 3D y texturas. Este puerto está conectado directamente al microprocesador y la memoria a través del bus a una frecuencia más alta que los demás dispositivos, cabe señalar que este tipo de ranura de expansión sólo se encuentra
disponible para las tarjetas de video.



Tarjeta de video AGP.

Tarjetas de expansión.

Dentro de la PC podemos encontrar otros dispositivos instalados, como un módem (ya sea interno o externo), tarjeta de sonido, tarjeta de red, tarjeta SCSI para dispositivos de este tipo,etcétera.

Tarjeta de video

Al igual que todas las tarjetas es un dispositivo que se conecta a la computadora y genera texto
e imágenes en la pantalla de un monitor, éste es responsable de la calidad del video.

Tarjeta de video AGP.

- ***Módem (Modulador Demodulador)***

Este dispositivo permite interaccionar a la PC con una línea telefónica, ya sea para poder enviar
un fax o poder conectarse a Internet, existen principalmente dos tipos de módem: módem
interno y el externo.

a) Módem interno

Éste va insertado en la ranura de expansión de la PC, es una tarjeta, tiene dos conectores, en
uno se conecta la línea telefónica y en el otro el teléfono, utiliza software de comunicaciones.

Módem interno.

b) Módem externo

Es un dispositivo externo, generalmente de forma rectangular, que se coloca en el exterior de la,PC, se conecta a través del puerto serial y utiliza un software de comunicaciones.

Módem externo.

- ***Tarjeta de sonido***

Es un dispositivo que permite a la computadora reproducir sonidos, hay diferentes formatos de sonidos, hoy en día son parte importante de un equipo de cómputo.

Tarjetas de sonido.