Sistema de Cómputo.

TEMA 1

Resumen del primer tema de la asignatura Fundamentos del Software,
Hecho por Marta Gómez Macías

TEMA 1: SISTEMA DE CÓMPUTO

DEFINICIONES BÁSICAS

- **Informática:** es el conjunto de conocimientos científicos y técnicas que hacen posible el tratamiento automático de la información por medio de ordenadores. La disciplina de informática es el cuerpo de conocimiento que trata del diseño, análisis, implementación, eficiencia y aplicación de procesos que transforman la información.
- **Computador:** es una máquina capaz de aceptar unos datos dados de entrada, efectuar con ellos operaciones lógicas y aritméticas, y proporcionar la información resultante a través de un medio de salida; todo ello sin intervención de un operador humano y bajo el control de un programa de instrucciones previamente almacenado en el propio computador. El computador actúa con dos tipos de informaciones: las *instrucciones*, que indican a la máquina qué es lo que tiene que hacer y los *datos*, que son los elementos que procesa o genera el programa.
- **Calculadora:** es una máquina capaz de efectuar operaciones aritméticas bajo el control directo del usuario.
- **Dato:** es un conjunto de símbolos utilizado para expresar o representar un valor numérico, un hecho, un objeto o idea. Las salidas de un programa se suelen denominar *datos de salida*.
- **Codificación:** es una transformación que representa los elementos de un conjunto mediante los de otro, de forma tal que a cada elemento del primer conjunto le corresponda un elemento distinto del segundo. Con los códigos se puede comprimir y estructurar la información.
- **Bit:** es la unidad más elemental de información. Este término procede de la contracción de las palabras *BInary* y *digiT*. Por tanto, es una posición o variable que toma el valor o ó 1.

  La información se representa por medio de caracteres y se codifica en un alfabeto binario; es decir, en bits. Por tanto a cada carácter le corresponde un cierto numero de bits.
- **Byte:** es un conjunto de 8 bits considerado como una unidad. El byte es una unidad pequeña por lo que se suelen usar múltiplos que son potencias enteras de 2.

MÚLTIPLOS			
Prefijo	**Símbolo**	**Factor decimal**	**Factor binario (nº de Bytes o bits)**
Kilo-	k-	$1000^1=10^3$	$1024^1=2^{10}=1.024$
Mega-	M-	$1000^2=10^6$	$1024^2=2^{20}=1_{1}048.576$
Giga-	G-	$1000^3=10^9$	$1024^3=2^{30}=2.073_{1}741.824$
Tera-	T-	$1000^4=10^{12}$	$1024^4=2^{40}=1_{2}099.511_{1}627.776$
Peta-	P-	$1000^5=10^{15}$	$1024^5=2^{50}=1.125_{2}899.906_{1}842.624$
Exa-	E-	$1000^6=10^{18}$	$1024^6=2^{60}=1_{3}152.921_{2}504.606_{1}876.976$
Zetta-	Z-	$1000^7=10^{21}$	$1024^7=2^{70}=1.180_{3}591.620_{2}717.411_{1}303.424$
Yotta-	Y-	$1000^8=10^{24}$	$1024^8=2^{80}=1_{4}280.925_{3}819.614_{2}629.174_{1}706.176$

- **Unidad de entrada:** dispositivo por el que se introducen los datos e instrucciones. En estas unidades se transforman las informaciones de entrada en señales binarias de naturaleza eléctrica. Son unidades de entrada: un teclado, un ratóm, un escáner, etc.
- **Unidad de salida:** dispositivo por el que se obtienen los resultados de los programas ejecutados en el computador. En estas unidades se transforman las señales eléctricas binarias en información perceptible por el usuario. Son dispositivos de salida: una pantalla, una impresora, etc.
- **Memoria interna:** unidad donde se almacenan tanto los datos como las instrucciones durante la ejecución de los programas. La memoria interna (también denominada *memoria central* o *memoria principal*) actúa con gran velocidad y está ligada a las unidades más rápidas del computador. Para que un programa se ejecute, debe estar almacenado (*cargado*) en la memoria principal. En los computadores actuales está formada por circuitos electrónicos integrados (chips).
    - La memoria está dividida en *posiciones* (también denominadas *palabras de memoria*) de un determinado número de bits, que es donde se almacena o memoriza la información. Cada palabra únicamente se puede referenciar por su dirección. Normalmente hay una zona de la memoria en la que sólo se puede leer (*Memoria ROM*) y que es permanente, y otra en la que se puede leer y escribir (*Memoria RAM*) y que es volátil.
- **Memoria extrena:** Para guardar información se utilizan otros tipos de memoria como discos magnéticos, discos ópticos y cintas magnéticas, que son más lentos pero pero tienen mucha más capacidad que la memoria principal. El conjunto de estas unidades se *denomina memoria externa*, *memoria masiva* o *memoria secundaria*.
- **Unidad de tratamiento (PU, *Proccesing Unit*):** Como elemento principal contiene a la unidad aritmético-lógica o *ALU* (*Arithmetic Logic Unit*), que contiene los circuitos electrónicos con los que se hacen las operaciones de tipo aritmético y lógico. Esta unidad también suele denominarse *camino de datos* (o *ruta de datos*). Ya que aparte de contener a la ALU incluye otros elementos auxiliares por donde se transmiten (*buses de datos*), o *registros* para almacenar temporalmente los datos.
- **Registro:** pequeña memoria diseñada para almacenar un dato, instrucción o dirección de memoria.
- **Unidad de control (CU, *Control Unit*):** de acuerdo con el código de operación de la instrucción captada y con las *señales de estado* procedentes de los distintos elementos del computador, genera *señales de control* dirigidas a todas las unidades, ordenando las operaciones que implican la ejecución de la instrucción. Esta unidad contiene un *reloj* que es un *generador electrónico de pulsos* que sincroniza todas las operaciones elementales del computador. El período de esta señal se denomina *tiempo de ciclo* y está comprendido entre décimas de nanosegundos y varios microsegundos. La *frecuencia del reloj* (inverso del tiempo de ciclo) suele darse entre millones de ciclos/segundo (Megahercios) o en miles de millones de ciclos/segundo (Gigahercios).
    - Los distintos elementos de un computador se interconectan a través de conjuntos de hilos, líneas o pistas eléctricamente conductores que suelen llevar en un instante dado (*en paralelo*) la información completa de una instrucciónm un dato o una dirección.

- **Bus:** conjunto de conductores que transmite información del mismo tipo entre unidades distintas. El *ancho de bus* es el numero de hilos que contiene, o numero de bits que transmite simultáneamente, en paralelo.
- **Periféricos:** conjunto de unidades de E/S y de memoria externa de un computador.
- **Unidades centrales:** resto de unidades; es decir, memoria interna y unidades de control y ALU.
- **Unidad de procesamiento central (CPU, Central Processing Unit):** o sencillamente, *procesador*, es el conjunto de unidad de control y unidad de tratamiento.
    - El grado de miniaturización alcanzado en la integración de circuitos ha llegado a ser tan alto que en un único chip se pueden incluir todos los elementos de un procesador.
- **Microprocesador:** es un procesador (CPU) implantado en un circuito integrado. Por sí solo no realiza ninguna función; para funcionar adecuadamente debe estar interconectado a un conjunto de circuitos a los que controla o monitoriza.
- **Microcontrolador:** es un circuito integrado que contiene, total o parcialmente, los cinco elementos básicos de un computador completo (unidad de control, unidad de tratamiento, memoria y puertos de E/S). Se diferencian de los microprocesadores en que aparte del procesador contienen otros elementos y están orientados a aplicaciones específicas de control y suelen instalarse embebidos dentro del sistema que controlan.
- **Interfaz:** conjunto de elementos adaptadores que sirven de comunicación entre dos módulos. La *interfaz entre dos programas* e *interfaz de usuario* es el conjunto de instrucciones que hace que un programa o aplicación intercambie información con el usuario del mismo.

CAMBIO DE BASE: BINARIO, OCTAL, HEXADECIMAL

Un *Sistema de numeración en base b* utiliza para representar los números de un alfabeto compuesto por b símbolos o cifras. Así, todo número se expresa mediante un conjunto de cifras, contribuyendo cada una de ellas con un valor que depende de:

- La cifra en sí
- La posición que ocupe dentro del número.

Generalizando para cualquier base, se tiene que la representación de un número N en la base b:

$$N = \ldots n_4\, n_3\, n_2\, n_1\, n_0\, n_{-1}\, n_{-2} \ldots$$

En la siguiente tabla se representan los sistemas de numeración más usuales en informática; los dos últimos (octal y hexadecimal) se conocen como códigos intermedios ya que se usan como intermedio en la transformación de un número hexadecimal a binario:

Denominación	Base	Símbolos utilizados
Decimal	b = 10	$S_{10} = \{0,1,2,3,4,5,6,7,8,9\}$
Binario	b = 2	$S_2 = \{0,1\}$
Octal	b = 8	$S_8 = \{0,1,2,3,4,5,6,7\}$
Hexadecimal	b = 16	$S_{16} = \{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F\}$

OPERACIONES ARITMÉTICAS CON VARIABLES BINARIAS

Suma:

a	b	a+b
0	0	0
0	1	1
1	0	1
1	1	0 y me llevo 1

Resta:

a	b	a-b
0	0	0
0	1	1 y adeudo 1
1	0	1
1	1	0

Multiplicación:

a	b	a*b
0	0	0
0	1	0
1	0	0
1	1	1

División:

a	b	a/b
0	0	Indeterminado
0	1	0
1	0	Infinito
1	1	1

Al multiplicar por $10)_2$ equivale a añadir un cero a la derecha, siendo esto similar a multiplicar por $10)_{10}$ un número decimal. De la misma forma dividir por $2)_{10} = 10)_2$ se hace desplazando el punto decimal a la izquierda, o eliminando ceros a la derecha.

REPRESENTACIÓN EN COMPLEMENTOS

El *complemento a uno* de un número binario, N, es el número que resulta de sustituir unos por cero y ceros por uno en el número original.

El *complemento a dos* de un número binario, N, es el número que resulta de sumar uno al complemento a uno de dicho número.

TRANSFORMACIONES ENTRE BASES DISTINTAS

Para pasar de decimal a binario, y viceversa, se suele realizar utilizando como pasos intermedios la base octal o la hexadecimal:

Decimal → hexadecimal → binario

Binario → hexadecimal → decimal

Decimal → octal → binario

Binario → octal → decimal.

Podemos pasar de base 2 a base 10 directamente, por ejemplo:

$110100)_2 = (1*2^5) + (1*2^4) + (1*2^3) + (1*2^2) = 2^5 + 2^4 + 2^3 + 2^2 = 32 + 16 + 4 = 52)_{10}$

Solo tenemos que sumar los pesos de las posiciones en las que hay un 1.

Para la transformación inversa, de decimal a binario:

- La parte entera se obtiene dividiendo por 2 la parte entera del número decimal de partida y los cocientes que se vayan obteniendo. Los restos de estas divisiones y el último cociente (siempre serán ceros y unos) son las cifras binarias. El último cociente será el *bit más significativo (MSB, Most Significative Bit)* y el primer resto será el *bit menos significativo (LSB, Least Significative Bit)*
  - Ejemplo:
    - 26 |_2_
      6 13 |_2_
      0 **1** 6 |_2_
      0 3 |_2_
      1 **1**
      $26)_{10} = 11010)_2$
- La parte fraccionaria se obtiene multiplicando por 2 sucesivamente la parte fraccionaria del número decimal de partida y las partes fraccionarias que se van obteniendo en los productos sucesivos (serán ceros y unos).
  - Ejemplo: numero decimal = 0,1875
    - 0,1875 x 2 = **0**,37500
      0,375 x 2 = **0**,7500
      0,7500 x 2 = **1**,5000
      0,5000 x 2 = **1**,0000
      Luego $0,1875)_{10} = 0,00011)_2$

En la transformación de *octal a binario y de hexadecimal a binario* se pueden aplicar algoritmos análogos a los vistos.

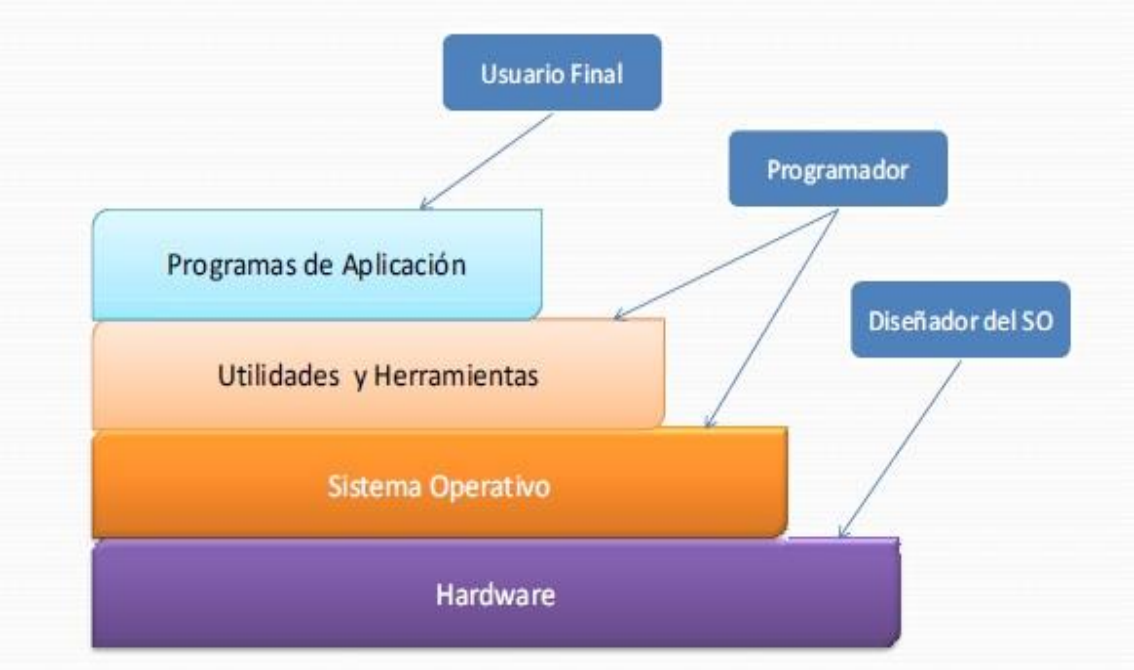
MÁS DEFINICIONES BÁSICAS

- **Hardware**: conjunto de los componentes que integran su parte material; es decir, el conjunto de circuitos electrónicos, cables, armarios, dispositivos electromecánicos y otros elementos físicos que forman el computador.
- **Software**: software o soporte lógico es el conjunto de programas (del sistema operativo, de utilidades y de los usuarios) ejecutables por el ordenador.
- **Firmware**: bloque de instrucciones de máquina para propósitos específicos, grabado en una memoria, normalmente de lectura/escritura (ROM, EEPROM, flash, etc.), que establece la lógica de más bajo nivel que controla los circuitos electrónicos de un dispositivo de cualquier tipo. Está fuertemente integrado con la electrónica del dispositivo siendo el software que tiene directa interacción con el hardware: es el encargado de controlarlo para ejecutar correctamente las instrucciones externas. En resumen, un firmware que maneja físicamente al hardware.

- **Sistema operativo**: programa que controla la ejecución de aplicaciones y programas y que actúa de interfaz entre las aplicaciones y el hardware. Sus objetivos son facilitar el uso del computador, gestionar sus recursos de manera eficiente y además permitir el desarrollar y probar nuevas funciones en el sistema sin interferir en su servicio.

SISTEMA OPERATIVO COMO INTERFAZ DE UN COMPUTADOR

Esta es la relación jerárquica en la que podemos ver al hardware y al software utilizados para proporcionar aplicaciones a los usuarios:



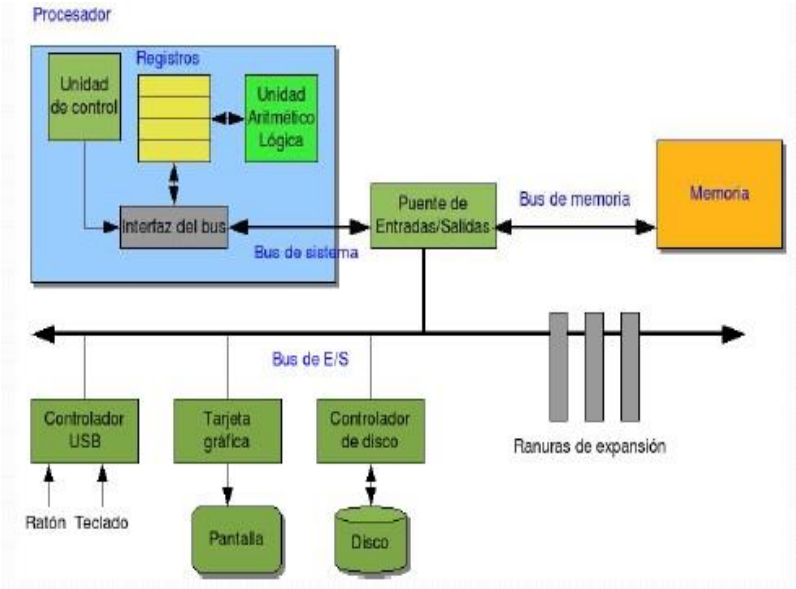
El usuario final no suele preocuparse del hardware del computador, por tanto, el usuario final ve un sistema de computación en términos de un conjunto de aplicaciones. Una aplicación se puede expresar en un lenguaje de programación y normalmente es desarrollada por un programador. Éste la desarrolla usando un conjunto de programas del sistema conocidos como utilidades y que lo asisten en las fases de creación de programas, gestión de ficheros y control de dispositivos E/S. Luego las aplicaciones invocarán también a estas utilidades en su ejecución para realizar distintas tareas. El programa más importante es el sistema operativo. Éste oculta los detalles del hardware al programador y le proporciona una interfaz apropiada para utilizar el sistema, es decir, actúa como mediador.

7 Sistema de Cómputo.

CAPA HARDWARE

ARQUITECTURA DE UN SISTEMA

La arquitectura de un sistema es:



Consultar el apartado de definiciones básicas para saber qué es cada cosa.

REGISTROS DEL PROCESADOR

Un procesador incluye un conjunto de registros que proporcionan un tipo de memoria que es más rápida y de menos capacidad que la memoria principal. Tienen dos funciones:

- **Registros visibles para el usuario:** Permiten al programador en lenguaje máquina minimizar las referencias a memoria principal optimizando el uso de registros. Para lenguajes de alto nivel, un compilador que realice optimización intentará tomar decisiones inteligentes sobre qué variables se asignan a registros y cuáles a posiciones de memoria principal.
- **Registros de control y estado:** usador por el procesador para controlar su operación y por rutinas privilegiadas del sistema operativo para controlar la ejecución de programas.

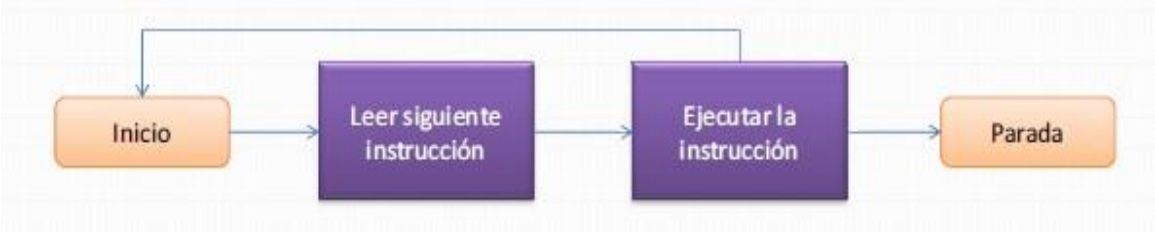
A un registro visible para el usuario se puede acceder por medio del lenguaje de máquina ejecutado por el procesador que está generalmente disponible para todos los programas. Los tipos de registros disponibles suelen ser de datos, dirección o de códigos de condición. El programador puede usar los *registros de datos* con cualquier instrucción máquina que realice operaciones sobre ellos, aunque suele haber restricciones. Los *registros de dirección* contienen direcciones de memoria principal de datos e instrucciones o una parte de la dirección que se usa para el cálculo de la dirección completa y se dedican

a direccionamiento de memoria. Como ejemplos de registros de dirección tenemos el *registro índice* (implica sumar un índice a un valor de base para obtener una dirección efectiva), *el puntero de segmento* (divide la memoria en segmentos de longitud variable, cada referencia a memoria consta de una referencia a un determinado segmento y un desplazamiento dentro del segmento) y el *puntero de pila* (que permite el uso de instrucciones que no contienes campo de dirección).

Los registros de control y estado se emplean para controlar el funcionamiento del procesador. En la mayoría de máquinas no son visibles para el usuario, a algunas se puede acceder mediante instrucciones máquina ejecutadas en modo de control o de sistema operativo. Algunos de estos registros son el *Contador de programa (Program Counter, PC)* que contiene la dirección de la próxima instrucción que se leerá de la memoria; el *Registro de instrucción (Instruction Registrer, IR)* que contiene la última instrucción leída; la *Palabra de estado del programa (Program Status Word, PSW)* que contiene información de estado, *códigos de condición (o indicadores)* que son bits cuyo valor es asignado por el hardware de procesador teniendo en cuenta el resultado de las operaciones, o bits para habilitar/inhabilitar interrupciones o un bit de modo usuario/supervisor.

EJECUCIÓN DE INSTRUCCIONES

En su forma más simple, el procesamiento de una instrucción consta de dos pasos: el procesador lee (busca) instrucciones de la memoria, una cada vez, y ejecuta cada una de ellas. La ejecución del programa consiste en repetir el proceso de búsqueda y ejecución de instrucciones. El *ciclo de instrucción* es el procesamiento requerido por una única instrucción, en la figura siguiente se describe el ciclo de instrucción utilizando la descripción simplificada de dos pasos (*fase de búsqueda y de ejecución*). La ejecución se detiene sólo si se apaga la máquina, se produce un error irrecuperable o se ejecuta una instrucción del programa que para el procesador.



Al principio de cada ciclo de instrucción, el procesador lee una instrucción de la memoria. El contador de programa almacena la dirección de la siguiente instrucción que se va a leer y, a menos que se le indique otra cosa, el procesador siempre incrementa el contador de programa después de cada instrucción ejecutada, de manera que se leerá la siguiente instrucción en orden secuencial. La instrucción leída se carga dentro de un registro del procesador conocido como registro de instrucción, esta instrucción contiene bits que especifican la acción que debe hacer el procesador y éste tras interpretarla la lleva a cabo. Estas acciones suelen dividirse en cuatro categorías:

- **Procesador-memoria**: se pueden transferir datos desde el procesador a la memoria o viceversa.
- **Procesador-E/S:** se pueden enviar datos a un dispositivo periférico o recibirlos desde el mismo, transfiriéndolos entre el procesador y un módulo de E/S.

- **Procesamiento de datos:** el procesador puede realizar algunas operaciones aritméticas o lógicas sobre datos.
- **Control:** una instrucción puede especificar que se va a alterar la secuencia de ejecución.

COMUNICACIONES DE E/S

Hay tres técnicas para llevar a cabo las operaciones de E/S:

- E/S programada
- E/S dirigida de interrupciones
- Acceso directo a memoria (*Direct Memory Access*, DMA)

Cuando el procesador ejecuta un programa y encuentra una instrucción relacionada con E/S, ejecuta esa instrucción generando un mandato al módulo de E/S apropiado. En el caso de la E/S programada, el módulo realiza la acción apropiada y fija los bits correspondientes en el registro estado, pero no realiza ninguna acción para avisar al procesador. Por tanto, este debe tomar un papel activo para determinar cuándo se completa la instrucción de E/S comprobando periódicamente el estado del módulo de E/S. El procesador es responsable de extraer los datos de memoria principal en una operación de salida y almacenarlos en ella en una operación de entrada, el software de E/S se escribe de manera que el procesador ejecuta instrucciones que le dan control directo de la operación de E/S, por tanto el juego de instrucciones incluye instrucciones de E/S de *control* (para activar un dispositivo externo y especificarle qué debe hacer), de *estado* (para comprobar condiciones de estado asociadas a un módulo de E/S y sus periféricos) y de *transferencia* (para leer o/y escribir datos entre los registros del procesador y dispositivos externos). La desventaja de la E/S programada es que el procesador tiene que esperar mucho hasta que el módulo de E/S esté listo para la salida o recepción de datos y mientras debe estar comprobando periódicamente el estado, por lo que el nivel de rendimiento del sistema de degrada.

Una alternativa sería que el procesador generase un mandato de E/S para un módulo y siga realizando algún otro trabajo útil, el módulo de E/S interrumpirá más tarde al procesador para solicitar su servicio cuando esté listo para intercambiar datos con el mismo. El procesador ejecutará la transferencia de datos, como antes, y después reanudará el procesamiento previo. Así funciona la E/S dirigida por interrupciones que es más eficiente que la E/S programada ya que elimina la espera innecesaria, sin embargo, la E/S dirigida por interrupciones aún consume mucho tiempo de procesador ya que cada palabra de datos que va desde la memoria la módulo de E/S o viceversa debe pasar a través del procesador.

Para transferir grandes volúmenes de datos se requiere una técnica más eficiente: el *acceso directo a memoria (Direct Memory Access, DMA),* cuya función puede llevarla a cabo un módulo separado conectado en el bus del sistema o estar incluida en el módulo de E/S. Cuando el procesador quiere leer o escribir un bloque de datos, genera un mandato al módulo de DMA enviándole si se trata de una lectura o de una escritura, la dirección del dispositivo involucrado, la posición inicial de memoria en la que se desea leer o escribir datos y el número de palabras que se pretenden leer o escribir. Después el procesador continúa con otro trabajo y esta operación de E/S la hará el módulo de DMA que transferirá el bloque completo de datos hacia memoria o desde ella sin pasar por el procesador. Así, el procesador

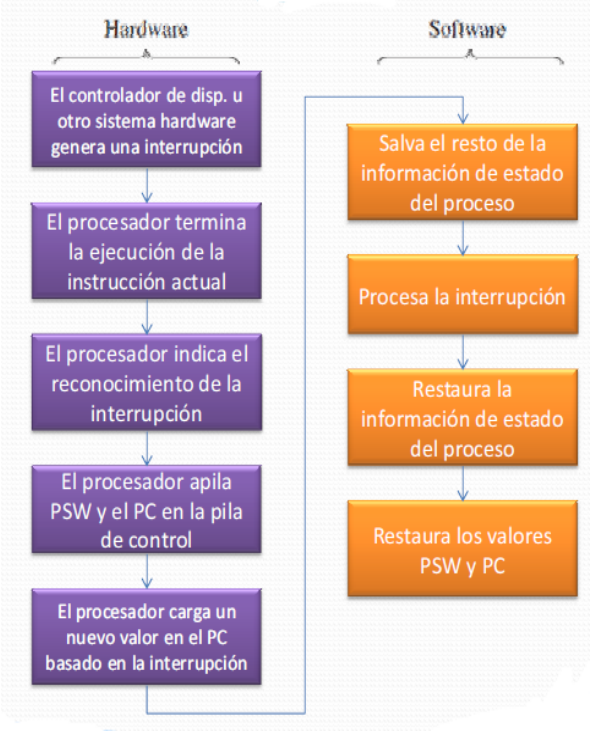
solo está involucrado al principio y al final de la transferencia. Aunque como el DMA necesita usar el bus, si el procesador también lo necesita debe esperar al módulo de DMA. Esto solo hace que el procesador ejecute más lentamente durante una transferencia, aunque aun así, el DMA es mucho más eficiente que la E/S dirigida por interrupciones.

PROCESAMIENTO DE INTERRUPCIONES

Cuando aparece una interrupción se produce la siguiente secuencia de eventos en el hardware:

1. El dispositivo genera una señal de interrupción hacia el procesador
2. El procesador termina la ejecución del a instrucción actual antes de responder a la interrupción, tal y como se indica en el ciclo de instrucción:

3. El procesador comprueba si hay una petición de interrupción pendiente, determina que hay una y manda una señal de reconocimiento al dispositivo que produjo la interrupción. Este reconocimiento permite que el dispositivo elimine su señal de interrupción.
4. En ese momento, el procesador necesita pararse para transferir el control a la rutina de interrupción. Necesita la información para reanudar el programa en el momento de la interrupción, *palabra de estado del programa (PSW)* y la posición de la siguiente instrucción que se va a ejecutar, contenida en el contador de programa.
5. A continuación, el procesador carga el contador del programa con la posición del punto de entrada de la rutina de manejo de interrupción que responderá a esta interrupción. Si hay más de una rutina de manejo de interrupción, el procesador debe determinar cuál invocar. Una vez cargado el contador de programa, el procesador continúa con el siguiente ciclo de instrucción, que comienza con una lectura de instrucción. Esta lectura está determinada por el contenido del contador del programa, el resultado es que se transfiere el control al programa manejador de interrupción. Su ejecución conlleva:
6. El contador de programa y la PSW vinculados con el programa interrumpido se han almacenado en la pila del sistema. Sin embargo, hay otra información que se considera parte del estado del programa en ejecución, se necesita salvar el contenido de los registros del



procesador ya que estos registros los podría utilizar el manejador de interrupciones por lo que se deben guardar estos valores y cualquier información de estado.

7. El manejador de interrupción puede empezar a procesar la interrupción que incluye un examen de la información de estado relacionada con la operación de E/S o con otro evento distinto que haya causado la interrupción. También puede implicar el envío de mandatos adicionales o reconocimientos al dispositivo de E/S.
8. Cuando se completa el procesamiento de la interrupción, se recuperan los valores de los registros salvados en la pila y se restituyen en los registros.
9. La última acción consiste en restituir de la pila los valores de la PSW y del contador del programa. Como resultado, la siguiente instrucción que se va a ejecutar corresponderá al programa previamente interrumpido.

Es importante salvar la información de estado del programa interrumpido para su posterior reanudación, ya que la interrupción no es una rutina de llamada desde el programa y su aparición es imprevisible.

PROTECCIÓN

El procesador dispone de dos modos de ejecución de instrucciones:

- **Modo usuario:** es el nivel menos permisivo en el que la computadora ejecuta solamente un subconjunto de las instrucciones máquina, quedando prohibidas las demás. También queda prohibido el acceso a determinados registros, o partes de esos registros, y a determinadas zonas del mapa de memoria y de E/S.
- **Modo kernel/supervisor:** es el nivel más permisivo en el que la computadora ejecuta todas las instrucciones sin ninguna restricción y permite el acceso a todos los registros y mapas de direcciones.

Uno o varios bits del registro de estado establecen el nivel en el que está ejecutando la máquina, modificando estos bits, se cambia de nivel de ejecución. Estos niveles se incluyen para dar soporte al sistema operativo. Los programas de usuario, por nivel de seguridad no podrán realizar determinadas acciones a nivel de usuario, por su lado, el SO, que ejecuta en el nivel del núcleo, puede ejecutar todo tipo de acciones.

Uno de los objetivos prioritarios de protección son los periféricos, ya que prohibiendo el acceso directo de los usuarios a los periféricos se impide que puedan acceder a la información almacenada por otros usuarios en esos periféricos. Por esta razón, toda la E/S es accesible solamente en nivel de núcleo, nivel que debe estar reservado para el sistema operativo.

Para evitar que un programa de usuario pueda poner el procesador en nivel de núcleo, no existe ninguna instrucción máquina que realice este cambio, sin embargo, existe la instrucción máquina inversa que cambia de nivel núcleo a nivel usuario. Esta instrucción es utilizada por el SO antes de dejar que se ejecute un programa de usuario. Se puede argumentar que la instrucción TRAP realiza el cambio de nivel de ejecución de usuario a núcleo, sin embargo, el objetivo de esta instrucción es producir una interrupción, como efecto indirecto de la interrupción, se hace un cambio de nivel de ejecución y se deja de ejecutar el programa que incluye la instrucción de TRAP, pasándose a ejecutar el SO.

Los *mecanismos de protección de memoria* deben evitar que un programa en ejecución direccione posiciones de memoria que no le hayan sido asignadas por el sistema operativo. Una solución para sistemas sin memoria virtual es incluir una pareja de registros valla (límite y base), la zona de memoria contigua que se le asigna al programa estará entre esos límites. Todos los direccionamientos se calculan sumando el contenido del registro base de forma que para el programa es como si estuviese cargado a partir de la posición "o" de memoria, también se compara el valor de la dirección con el valor del límite y en caso de desbordamiento el comparador genera una excepción de violación de memoria para que el sistema operativo trate el tema, en la práctica, se parará el programa produciendo un *volcado de memoria (core dump)*. En los sistemas con memoria virtual hay dos mecanismos de protección de memoria: a nivel de usuario el procesador no permite acceder más que a una parte del mapa de memoria. A nivel de núcleo se puede direccionar todo el mapa de memoria virtual mientras que en nivel de usuario solo se direcciona una parte del mapa. También se puede dotar al procesador de un *registro RIED (registro de identificador de espacio de direccionamiento),* así se consigue que cada programa en ejecución disponga de su propio espacio virtual, por lo que no puede acceder a los espacios de memoria de otros procesos. Otro mecanismo se basa en la tabla de páginas:

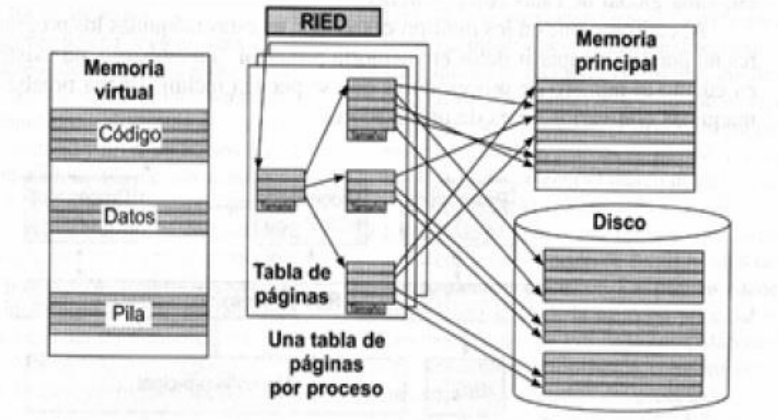


Figura 1.26. *La tabla de páginas como mecanismo de protección de memoria.*

La tabla de páginas de cada programa en ejecución se selecciona mediante el RIED y especifica los espacios de memoria asignados por el SO a ese

programa. La *memory management unit (MMU),* al mismo tiempo que realiza la traducción de cada dirección , comprueba que no se sobrepase el límite de ninguna de las tablas involucradas y en caso de sobrepasarse, genera una excepción de violación de memoria para que el SO realice la acción correcta oportuna.

SISTEMA OPERATIVO

El sistema operativo proporciona normalmente servicios en las siguientes áreas:

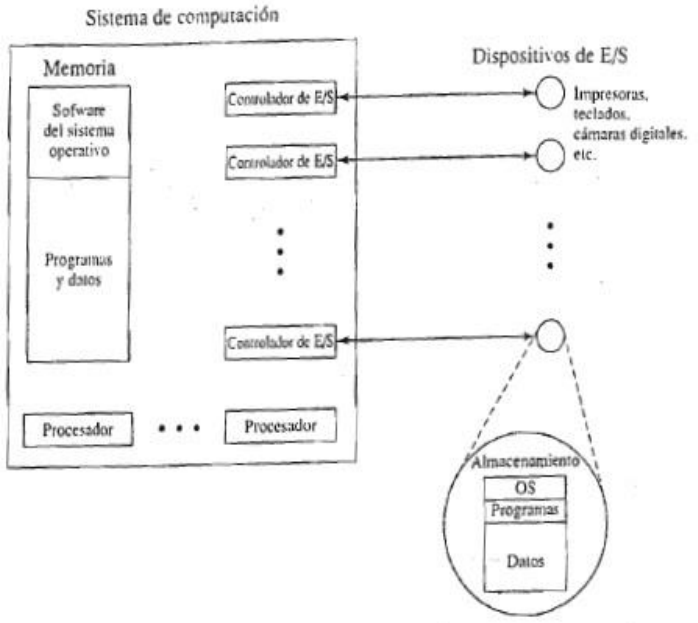
- **Desarrollo de programas:** proporciona una variedad de utilidades y servicios, como editores y depuradores, para asistir al programador en la creación de programas, estos servicios se suelen ofrecer como utilidades que aunque no forman parte del núcleo se ofrecen con dicho sistema y se conocen como herramientas de desarrollo de aplicación.
- **Ejecución de programas:** se necesita realizar una serie de pasos para ejecutar un programa. Las instrucciones y datos se deben cargar en memoria principal, los dispositivos de E/S y los ficheros se deben inicializar y otros recursos deben prepararse. Los SO realizan estas labores de planificación en nombre del usuario.
- **Acceso a dispositivos de E/S:** cada dispositivo de E/S requiere su propio conjunto peculiar de instrucciones o señales de control para cada operación. El sistema operativo proporciona una interfaz uniforme que esconde esos detalles de forma que los programadores puedan acceder a dichos dispositivos utilizando lecturas y escrituras sencillas.
- **Acceso controlado a los ficheros:** para el acceso a los ficheros, el sistema operativo debe reflejar una comprensión detallada no sólo de la naturaleza del dispositivo de E/S sino también de la estructura de los datos contenidos en los ficheros del sistema de almacenamiento. Adicionalmente, en el caso de un sistema multiusuario, el sistema operativo puede proporcionar mecanismos de protección para controlar el acceso a ficheros.
- **Acceso al sistema:** para sistemas compartidos o públicos, el SO controla el acceso al sistema completo y a recursos del sistema específicos. La función de acceso debe proporcionar protección a los recursos y a los datos evitando el uso no autorizado de los usuarios y resolviendo conflictos en el caso de conflicto de recursos.
- **Detección y respuesta a errores:** se pueden dar gran variedad de errores durante la ejecución de un sistema operativo. Éstos incluyen errores de hardware internos y externos y diferentes errores de software. En casa caso el sistema operativo debe proporcionar una respuesta que elimine la condición de error suponiendo el menor impacto en las aplicaciones que están en ejecución.
- **Contabilidad:** un buen sistema operativo recogerá estadísticas de uso de los diferentes recursos y monitorizará parámetros de rendimiento tales como el tiempo de respuesta. En cualquier sistema esta información es útil para anticipar las necesidades de mejoras futuras y para optimizar el sistema a fin de mejorar su rendimiento. En sistemas multiusuario se puede usar para facturar a los diferentes usuarios.

EL SISTEMA OPERATIVO COMO GESTOR DE RECURSOS

Gestionando los recursos del computador el SO tiene el control de las funciones básicas del mismo, pero tiene un mecanismo de control inusual en dos aspectos:

- Las funciones del sistema operativo actúan de la misma forma que el resto del software; es decir, se trata de un programa o conjunto de programas ejecutados por el procesador.
- El sistema operativo frecuentemente cede el control y depende del procesador para volver a retomarlo.

De hecho, el sistema operativo es un conjunto de programas que proporciona instrucciones para el procesador. La diferencia es que el sistema operativo tiene como objetivo dirigir al procesador en el uso de los recursos del sistema y en la temporización de la ejecución de otros programas. Para que el procesador pueda realizar esto, el sistema operativo debe dejar paso a la ejecución de otros programas y por tanto el sistema operativo deja el control para que el procesador pueda realizar trabajo "útil" y de nuevo retoma el control para permitir al procesador que realice la siguiente pieza de trabajo.



En la imagen se muestra los principales recursos gestionados por el sistema operativo. Una porción del sistema operativo se encuentra en la memoria principal. Esto incluye el *kernel* que contiene las funciones del sistema operativo más utilizadas y otras porciones del sistema operativo actualmente en uso. El resto de la memoria principal contiene programas y datos de usuario. La asignación de memoria principal es controlada de forma conjunta por el SO y el hardware de gestión de memoria del procesador. El SO decide cuándo un programa en ejecución puede utilizar un dispositivo de E/S y controla el acceso y uso de los ficheros. El procesador también es un recurso y el SO debe determinar cuánto tiempo de procesador debe asignarse a la ejecución de un programa de usuario particular. En el caso de un sistema multiprocesador, esta decisión debe ser tomada por todos los procesadores.

CARACTERÍSTICAS DESEABLES EN UN SISTEMA OPERATIVO

Podemos considerar que un sistema operativo tiene los siguientes tres objetivos:

- **Facilidad de uso:** un sistema operativo facilita el uso de un computador.
- **Eficiencia:** un sistema operativo permite que los recursos de un sistema de computación se puedan utilizar de manera eficiente.
- **Capacidad para evolucionar.** Un sistema operativo se debe construir de tal forma que se puedan desarrollar, probar e introducir nuevas funciones en el sistema sin interferir con su servicio. Un sistema operativo debe evolucionar en el tiempo por las siguientes razones:
    - **Actualizaciones de hardware más nuevos tipos de hardware:** por ejemplo la incorporación de páginas de páginas o el uso de terminales gráficos que permiten varias aplicaciones a la vez, requieren una gestión más sofisticada por parte del sistema operativo.
    - **Nuevos servicios:** en respuesta a la demanda, el SO debe ofrecer nuevos servicios. Como por ejemplo, actualizaciones en el SO para que soporte ciertos programas.
    - **Resolución de fallos:** cualquier SO tiene fallos. Estos fallos se descubren con el trascurso del tiempo y se resuelven, esto puede implicar la introducción de nuevos fallos.

PROGRAMAS DE SERVICIO DEL SO.

Son un conjunto de programas de servicio que pueden considerarse una ampliación del sistema operativo. Incluye programas para realizar tareas tales como *compactación de discos* o reubicación de archivos dentro de un disco para poder acceder a ellos más rápidamente, compactando los fragmentos libres de disco, *comprensión de datos*, o reducción del tamaño de un archivo utilizando algoritmos de compresión, *gestión de comunicaciones* a través de tarjeta de red o módem, *visualizadores y navegadores de internet*, programas para *respaldo de seguridad*, o copia del contenido de un disco fijo en cinta magnética, disquete o CD, *recuperación de archivos* erróneamente borrados, *antivirus*, *salvapantallas* que evitan imágenes fijas durante largos períodos de tiempo que pueden deteriorar la pantalla, etc. También se incluyen aquí herramientas generales que facilitan la construcción de las aplicaciones de los usuarios, sea cual sea la naturaleza de éstas, tales como *intérpretes*, *compiladores*, *editores de texto* y *cargadores/montadores*.