****

Día 1

****

Conceptos generales

Un sistema operativo es simplemente un conjunto de programas de software de propósito general con el fin de:
• Administrar los recursos del sistema
• Gestionar la correcta ejecución de varios programas (aplicaciones) de uno o varios usuarios
• Proveer una “visión” unificada a los usuarios y programadores mediante una capa de abstracción del hardware

Entre el hardware y el software de aplicación se encuentra el **sistema operativo**. El sistema operativo es un programa que establece la comunicación entre las distintas partes del hardware como: la plaqueta de video, la plaqueta de sonido, la impresora, la plaqueta principal (motherboard) y las aplicaciones

Clasificación de sistemas operativos:

Sistema de tiempo real

Se utilizan para sistemas de control industrial, centrales de conmutación, instrumentación científica, etc.
La interface con el usuario suele ser pobre
Su fortaleza consiste en administrar los recursos de la computadora para poder ejecutar una operación en particular en la misma cantidad de tiempo cada vez que ocurre en evento que la dispara. Es decir se busca evitar comportamientos aleatorios mediante:
• Máximo asegurado de duración de una interrupción
• Swapping reducido à Tareas con bajo consumo de memoria

La variación de velocidad y o respuesta un recurso no impacta en los resultados que entrega el sistema

Sistema de monotarea-monousuario

No pueden ejecutar mas de una tarea en forma concurrente
Transfieren todo el control del sistema a la aplicación que va a ejecutarse y esta tiene el control hasta que lo devuelve al sistema operativo.
El sistema de este tipo mas difundido fue el DOS “palms”

Sistema de Multitarea-monousuario

Utilizados hoy en día en las Pc’s de escritorio
Pueden trabajar con varios programas en memoria en forma estable dentro de un entorno de protección
Solo poseen interfaz para un usuario aunque pueden trabajar con varias sesiones.
Ejemplo Winxp home

Sistema de Multiusuario

Es la forma actual de los sistemas operativos, y en la que fueron pioneros los sistemas UNIX
Puede ejecutar diferentes sesiones en forma simultánea con múltiples tareas por sesión.
Ejemplos, Windows Server, Linux, unix,

Funciones del SO:

Gestión del procesador

Asignación de slots de tiempo de CPU para cada tarea (scheduling de procesos)

Gestion de la memoria

Protección de recursos
Memoria virtual
Memoria Cache

Gestion de dispositivos E/S

Protección al acceso por Concurrencia
Acceso al Hardware
Manejo de Interrupciones

Gestión de almacenamiento

Organizar/distribuir la información

Interfaz de Aplicaciones

Systems Call… y llamadas por codificación de interrupción

Interfaz de usuario

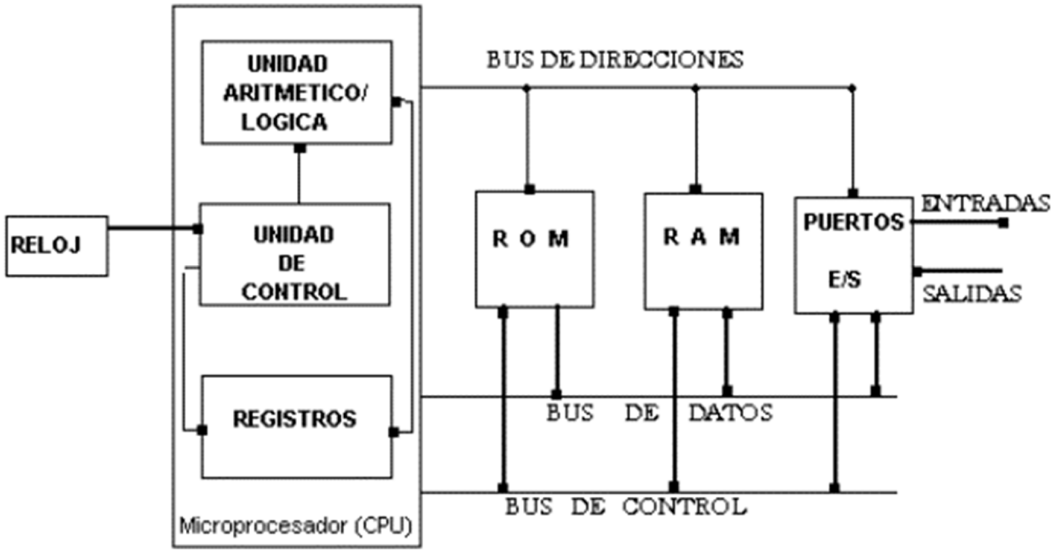
Consola de comandos
GUI
con propositos de administracion del SO

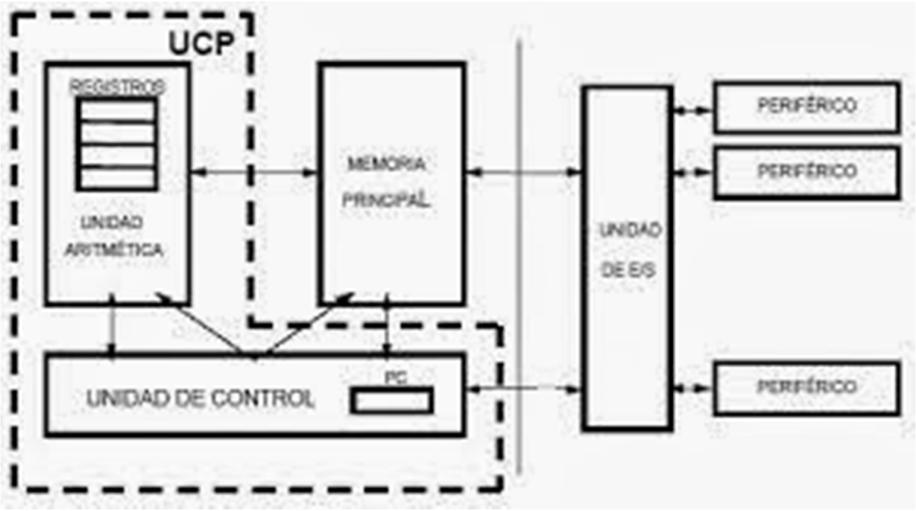
****

Día 2

****

Organización de computadores:





Como transfiere la información

-Bus de datos: Es el encargado de llevar la información a los diferentes dispositivos del sistema (Todo)

-Bus de direcciones: Es un bus dedicado para el procesador para determinar la dirección de donde se encuentra el dato, la información es trasmitida por Bus de datos

-Bus de control: Es quien administra el uso y el acceso a los datos y sus direcciones. y es la unidad de control la encargada de evitar que existan colisiones de de información en los canales

La capacidad de la memoria que se puede direccionar depende de la cantidad de bits que conforman el bus de direcciones, siendo 2^n el tamaño máximo en bits del banco de memoria que se podrá direccionar con n líneas. Por ejemplo, para direccionar una memoria de 256 bits, son necesarias al menos 8 líneas, pues $2^8 = 256$

Modelos de Transferencia de Información

- **E/S programada**

Los datos se intercambian entre el CPU y el módulo de E/S. El CPU ejecuta un programa que controla directamente la operación de E/S, incluyendo la comprobación del estado del dispositivo, el envío de la orden de lectura o escritura y la transferencia del dato. Cuando el CPU envía la orden debe esperar hasta que la operación de E/S concluya. Si el CPU es más rápido, éste estará ocioso. El CPU es el responsable de comprobar periódicamente el estado del módulo de E/S hasta que encuentre que la operación ha finalizado.

Normalmente habrá muchos dispositivos de E/S conectados al sistema a través de los módulos de E/S. Cada dispositivo tiene asociado un identificador o dirección. Cuando el CPU envía una orden de E/S, la orden contiene la dirección del dispositivo deseado.

- **E/S mediante interrupciones**

El problema con E/S programada es que el CPU tiene que esperar un tiempo considerable a que el módulo de E/S en cuestión esté preparado para recibir o transmitir los datos. El CPU debe estar comprobando continuamente el estado del módulo de E/S. Se degrada el desempeño del sistema.

Una alternativa es que el CPU tras enviar una orden de E/S continúe realizando algún trabajo útil. El módulo de E/S interrumpirá al CPU para solicitar su servicio cuando esté preparado para intercambiar datos. El CPU ejecuta la transferencia de datos y después continua con el procesamiento previo.

Procesamiento de la Interrupción

Cuando un dispositivo de E/S termina una operación de E/S, se produce la siguiente secuencia de eventos:

- **El dispositivo envía una señal de interrupción al procesador**
- **El procesador termina la ejecución de la instrucción en curso antes de responder a la interrupción.**
- **El procesador comprueba si hay alguna interrupción. Si hay alguna, envía una señal de reconocimiento al dispositivo que la originó**

- **El procesador debe prepararse para transferir el control a la rutina de interrupción. Debe guardar la información necesaria para continuar con el proceso en curso en el punto en que se interrumpió. Guarda en la pila del sistema el contenido de los registros, etc.**
- **El procesador carga en el PC la dirección de inicio del programa de gestión o servicio de interrupción solicitada.**
- **Una vez modificado el PC, el procesador continúa con el ciclo de instrucción siguiente. Es decir, se transfiere el control a la rutina servidora de la interrupción.**
- **Cuando finaliza el servicio de la interrupción, se restauran los valores de los registros.**

- **DMA(Direct Memory Access)**

Cuando un dispositivo tiene un bloque de datos preparado para enviar a la memoria, envía una petición al DMA poniendo una señal DRQn a "1". Si el canal de DMA se halla disponible, el DMA enviará una señal HRQ (hold request) al microprocesador. El microprocesador responderá dejando los buses libres y enviando una señal HLDA (hold acknowledge) al DMA. Luego el DMA obtiene el control de los buses poniendo la señal AEN a nivel alto y envía la dirección de memoria a ser escrita. Después el DMA envía la señal de DACKn (DMA acknowledge) al dispositivo. Finalmente el controlador de DMA se ocupa de manejar las señales de MEMW y IOR del bus de control. Cuando la transferencia de datos se ha completado vuelve a poner la señal HRQ a nivel bajo y el procesador recupera el control de los buses de nuevo.

Hay en total 256 interrupciones, de la 0 a la 7 (excepto la 5) son generadas directamente por el procesador. Las 8 a 0Fh son interrupciones por hardware primitivas de las PC. Desde la AT en adelante, se incorporó un segundo controlador de interrupciones que funciona en cascada con el primero a través de la interrupción 2 (de ahí que en la tabla siguiente se la denomine múltiplex). Las 8 interrupciones por hardware adicionales de las AT se ubican a partir del vector 70h.

Que son las IRQ: Interrupt Request (Pedido de Interrupción)
En los PCs, un IRQ es una señal de un dispositivo de hardware (por ej. el teclado o tarjeta de sonido) indicando que el dispositivo necesita que la CPU haga algo.

IRQ 0 (Temporalizador del sistema**)**
IRQ 1 (Teclado / controlador de teclado)
IRQ 2 (Cascada de irq 8 a 15)
IRQ 3(Segundo puerto serial COM2)
IRQ 4 (Primer puerto serial COM1)
IRQ 5 (Tarjeta de Sonido)
IRQ 6 (Controlador de Discos flexibles)

IRQ 7 (Primer Puerto paralelo LPT1)
IRQ 8 (Sistema reloj en tiempo real)
IRQ9 (No Tiene uso por defecto)
IRQ 10 (No tiene uso por defecto)
IRQ 11 Otros Usos Comunes
IRQ12 (PS/2 mouse.)
IRQ 14 (Primary IDE channel)
IRQ 15(Secondary IDE channel.)

Registros

Se emplean para controlar las instrucciones en ejecución del procesador , manejar direccionamiento de memoria, y propiciar capacidad aritmética

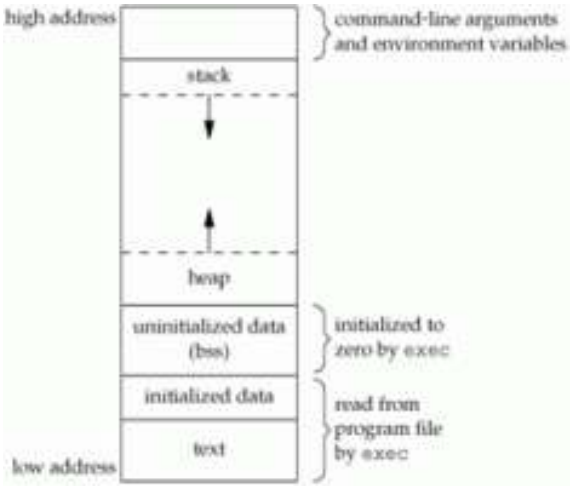
Tipos:

1. Segmento
2. Proposito general
3. Apuntadores
4. Banderas
5. Punteros de instrucción

Registros de Segmento:

Un registro de segmento contiene 16 bits de longitud y facilita un area de memoria para direccionamiento conocida como el segmento actual

Registro CS: almacena la dirección inicial del code segment a ejecutar
Registro DS:Almacena la dirección inicial de Data Segment
Registro SS:Almacena la dirección inicial de la pila (stack Segment)




```
#include <stdio.h>
```

```c
int temp_data = 100;

static int temp_bss;

void print_addr ( void )

{

int local_var = 100;

int *code_segment_address = ( int* ) &print_addr;

int *data_segment_address = &temp_data;

int *bss_address = &temp_bss;

int *stack_segment_address = &local_var;

printf ( "\nAddress of various segments:" );

printf ( "\n\tCode Segment : %p" , code_segment_address );

printf ( "\n\tData Segment : %p" , data_segment_address );

printf ( "\n\tBSS : %p" , bss_address );

printf ( "\n\tStack Segment : %p\n" , stack_segment_address );

}

int main ( )

{

print_addr ();

return 0;

}
```