RESUMEN TEMA 1 PSP

UNIDADES FUNCIONALES DEL PC:

Memoria Principal: Envía instrucciones a la **Unidad de Control (UC)** y envía y recibe datos a la **Unidad Aritmético Lógica (UAL).**

CPU:

- **Unidad de Control:** contiene la instrucción que se está ejecutando. La ejecución es atendida por el Kernel.
- **Unidad Aritmético Lógica:** almacena resultados de las operaciones y envía por el Bus los resultados.
- **Memoria Principal (RAM):** almacena instrucciones y datos necesarios en ejecución.
    - Selector de Memoria: posibilita la transferencia entre **RDM (Registro de direcciones de memoria)** y **RIM (Registro de Intercambio de Memoria)**
    - **RDM:** dirección desde la que se va a leer/escribir datos.
    - **RIM:** recibe el dato del RDM si es de lectura y lo escribe si es de escritura.
- **Bus del Sistema:** se encarga de la conexión y comunicación entre CPU y el resto de unidades.
    - **Líneas de Datos:** caminos físicos por donde se transmiten los datos.
    - **Líneas de Dirección:** se emplean para seleccionar la fuente o destino de la información que hay en el bus. Sólo transfiere direcciones.
    - **Líneas de Control:** gestionan el uso y acceso a los buses de datos.

INSTRUCCIONES:

Se componen de:

- **Código de Operación:** indica la operación que se realizará.
- **Operandos:** los valores que se usarán para la operación.

Tipos de Instrucciones:

- **Instrucciones de 3 operandos.**
- **Instrucciones de 2 operandos.**
- **Instrucciones de 1 operando.**
- **Instrucciones sin operandos.**

MÉTODOS DE DIRECCIONAMIENTO:

Un modo de direccionamiento es el modo que se utiliza en una instrucción para indicar la posición de memoria.

Tipos:

- **Direccionamiento Inmediato:** la instrucción contiene el dato que hay que emplear.
- **Direccionamiento Directo:** la instrucción contiene la dirección de memoria central.
- **Direccionamiento Indirecto:** la instrucción contiene la posición de memoria que contiene la dirección de ese dato.
- **Direccionamiento Relativo:** la dirección es calculada.

PROCESO:

Es un programa en ejecución, una instancia de un fichero ejecutable.

Un proceso implica:

- Su imagen de memoria, su espacio de direcciones de memoria donde se va a ejecutar. Cada proceso tiene asignado su propio espacio de direcciones de memoria.
- **Estado del Proceso:** conjunto de valores de los registros de la Unidad de Control y de la Unidad Aritmético Lógica. Valores de: registro de instrucción, registro de contador de programa, registros en la UAL.
- **Momento del Proceso:** podemos decir que es el valor del Registro Contador de Programa, contiene la dirección de memoria de la siguiente instrucción a ejecutarse.

Programa vs Proceso:

Un **Programa** es un fichero Binario Ejectuable en disco mientras un **Proceso** es una instancia de un porgrama en ejecución.

Un Porceso supone la existencia de:

- **Su propio espacio de direcciones** de memoria interna.
- **Valor de contador de programa** en un registro de la Unidad de Control.
- **El estado del proceso**: es el conjunto de valores de los distintos registros del procesador

ESTADOS DE UN PROCESO:

- **Estado Nuevo:** el proceso es creado (Sólo sucede una vez)
- **Estado Listo:** el proceso está en la cola de procesos listo para ejecutarse.
- **En Ejecución:** pasa desde el comienzo de la cola de procesos preparados e ejecutarse.
- **Estado Bloqueado:** el proceso está bloqueado esperando a que ocurra algún suceso. Desde aquí vuelve al Estado Listo antes de pasar a estar en ejecución.
- **Estado Terminado:** el proceso termina su ejecución o lo finaliza el proceso padre.

COLAS DE PROCESOS:

- **Cola de Procesos:** contiene todos los procesos existentes.
- **Cola de Procesos Preparados:** contiene todos los procesos en estado listo.
- **Cola de Dispositivos:** son las colas donde se encuentran los procesos en estado bloqueado a la espera de la demanda de alguna operación.

PLANIFICACIÓN DE PROCESOS:

Políticas de Planificación a Corto Plazo:

- **Planificación sin desalojo o cooperativa:** Sólo se cambia de proceso en ejecución si dicho proceso se bloquea o termina.
- **Planificación Apropiativa o con desalojo:** si desaloja un proce si aparece en la cola de procesos listos uno con mayor prioridad.
- **Tiempo Compartido:** se desaloja el proceso en ejecución porque finaliza su cuanto de ejecución.
    - **Cuanto de ejecución:** cantidad máxima de tiempo que un proceso puede estar en ejecución ininterrumpidamente.

Multiprogramación: ejecución concurrente de todos los procesos.

Políticas de Planificación a Largo Plazo:

- El planificador instancia o crea los nuevos procesos. Controla el númerod e procesos existentes y por lo tanto controla el grado de multiprogramación.

CAMBIO DE CONTEXTO:

El cambio de contexto es el cambio de un proceso a otro cuando se desaloja el que está en ejecución.

ÁRBOL DE PROCESOS:

Definimos un aplicación multiproceso como un árbol n-ario de procesos, donde todos los procesos son **cooperantes** ya que cooperan para la realización de una tarea común.

En general todos los procesos que se están ejecutando concurrentemente en un equipo, son procesos independientes que se ejecutan **asincrónicamente** y que no se comunican.

En una aplicación multiproceso puede ser **asincrónicos** si no se comunican o **sincrónicos** si se comunica.

La comunicación entre procesos padre-hijo es una forma de sincronización.

PROCESADOR CON VARIOS NÚCLEOS:

Cada núcleo puede atender a una instrucción pero del mismo proceso, del único proceso que está en ejecución, entonces es posible la multitarea; se ejecutan varios hilos pero de un mismo proceso.

MÉTODOS DE UN PROCESO:

- **.start():**
  - **Devuelve**: objeto de la clase Process
  - **Clase a la que pertenece:** ProcessBuilder
  - **Qué hace:** el nuevo proceso está en estado listo
- **.getInputStream():**
  - **Devuelve:** InputStream
  - **Clase a la que pertenece:** Process
  - **Qué hace:** realiza operaciones de lectura
- **.getOutputStream():**
  - **Devuelve:** OutputStream
  - **Clase a la que pertenece:** Process
  - **Qué hace:** realiza operaciones de escritura
- **.getErrorStream():**
  - **Devuelve:** InputStream
  - **Clase a la que pertenece:** Process
  - **Qué hace:** realiza operaciones de lectura de errores al padre
- **.destroy():**
  - **Devuelve:** void
  - **Clase a la que pertenece:** Process
  - **Qué hace:** finaliza el proceso hijo referenciado
- **.waitFor():**
  - **Devuelve:** int
  - **Clase a la que pertenece:** Process
  - **Qué hace:** bloquea al proceso padre hasta que el hijo finaliza
- **.exitValue():**
  - **Devuelve:** int
  - **Clase a la que pertenece:** Process
  - **Qué hace:** obtiene el valor de la operación exit del proceso hijo