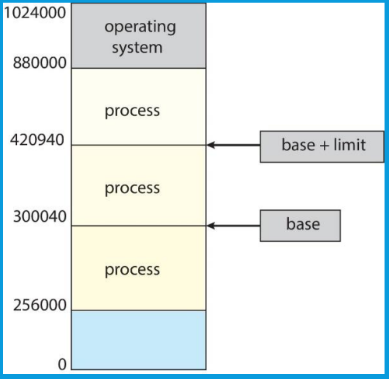
Memoria

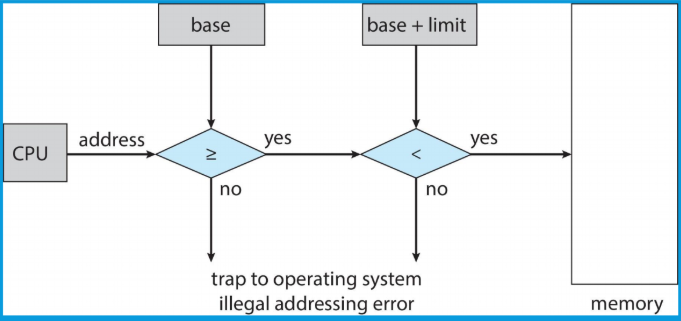
**Memoria y procesos**

* La memoria no distingue entre el código y la información.
* Las instrucciones e información tienen que cargarse de la memoria a los registros.
* Acceso a los registros usualmente toma un ciclo, acceso a la memoria puede tomar más.
* La memoria solo ve la solicitud de lectura y escritura en diferentes direcciones.

**Protección**

* Un proceso no debe modificar el espacio de memoria de otro proceso.
* Protección dada al checar los límites entre la base de la memoria y el límite.
* La memoria usada por el sistema operativo usualmente está en la parte superior o en la inferior
* 

# PROTECCIÓN DE MEMORIA DE HARDWARE



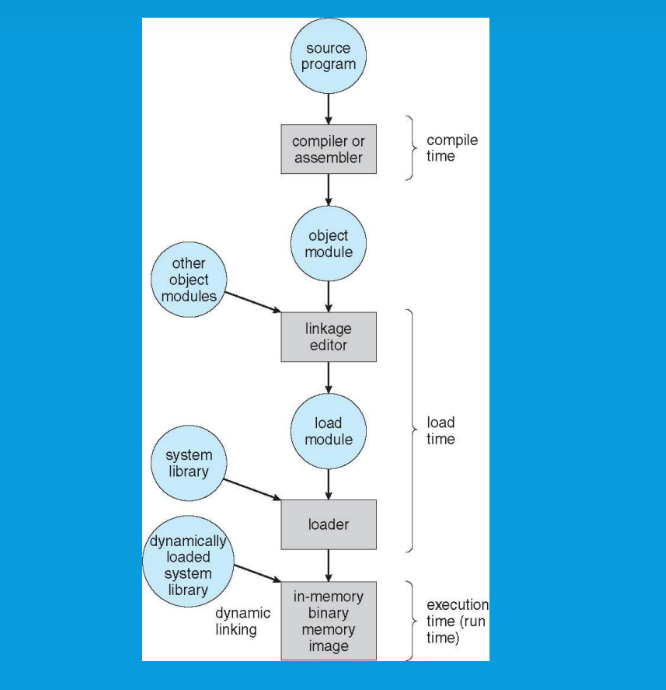
# **ASIGNACIÓN DE DIRECCIONES**

▪ Los procesos necesitan conocer la dirección para obtener instrucciones y datos.

▪ Las direcciones pueden ser estáticas si la dirección de carga base se conoce en el momento de la programación.

▪ Generalmente, las direcciones se pueden reubicar en el momento de la carga o la ejecución.

▪ La asignación de memoria a variables es en el tiempo de ejecución.



# **COMPILANDO, ENLACE Y CARGANDO**

▪ Al compilar, el programa se convierte en código objeto.

**▪ La vinculación** agrega las funciones de las bibliotecas y syscalls.

▪ La vinculación estática copia las bibliotecas en el ejecutable(Se carga desde el principio y se copia al ejecutable).

▪ La vinculación dinámica agrega un código auxiliar para llamar a la biblioteca. La dirección de la función cambia el código auxiliar.(se va cargando la librería como se va necesitando)

**▪ Cargando copia el programa en la memoria.**

▪ La carga estática copia todo el programa.

▪ Carga dinámica copia solo las funciones que se están utilizando.14:00

# **DIRECCIONES LÓGICAS vs FÍSICAS**

▪ Necesario para el código reubicable.

▪ La dirección física es la que está en la memoria principal

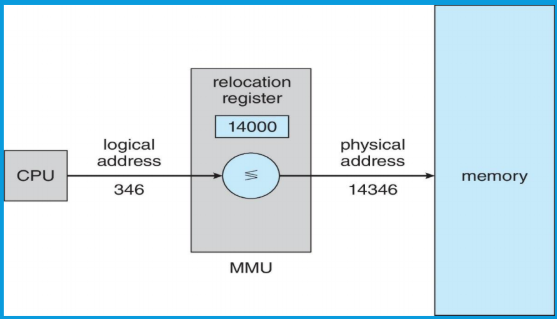
▪ La dirección lógica es la que ve el proceso.

▪ Cambios al ejecutar el programa.

▪ La MMU Unidad de Administración de Memoria se encarga de traducir las direcciones lógicas a físicas.Para conocer la dirección física solo sumamos la dirección lógica y la variable llamada relocation register. que está dentro del MMU.

▪ El proceso solo trata con direcciones lógicas.

▪ La dirección física suele ser más grande que la lógica



### **¿CÓMO SE ASIGNA LA MEMORIA?**

**Asignación contigua**

▪ Memoria en dos bloques, SO y usuario.

▪ Memoria de proceso asignada una después de la otra.

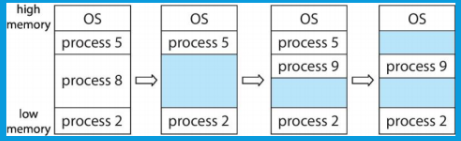
▪ Puede crecer dinámicamente

▪ Utiliza registros de reubicación para separar los procesos.

**Asignación variable(Es más común)**

▪ A cada proceso se le asigna memoria según sea necesario.

▪ El sistema operativo mantiene una lista de secciones asignadas como "agujeros".



### **PROBLEMA DE ASIGNACIÓN DE ALMACENAMIENTO DINÁMICO**

**▪ Primer ajuste(First fit)**

▪ Asignar el primer hoyo suficientemente grande

**▪ Mejor ajuste (Best fit)**

▪ Asignar el primer hoyo que sea lo suficientemente grande

**▪ Peor ajuste(Worst fit)**

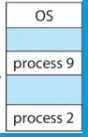
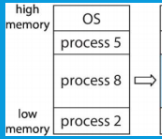
▪ Asignar el hoyo más grande

▪ El primer ajuste y el mejor ajuste tienen la mejor velocidad y utilización del almacenamiento.

### **PROBLEMA DE FRAGMENTACIÓN**

**▪ Fragmentación externa**

▪ Hay espacio para un nuevo proceso en la memoria, pero el espacio no es contiguo.



**▪ Fragmentación interna**

▪ La memoria asignada es más de lo necesario, lo que resulta en un desperdicio de memoria. En este se desperdicia espacio de memoria porque el programa ocupa más bloques pero no completamente.

▪ Estos problemas se resuelven mediante compactación, pero solo se pueden usar si la memoria se asigna dinámicamente.

### **Paginación (solución a la fragmentación externa)**

▪ Divida la memoria física en fragmentos del mismo tamaño llamados marcos.

▪ Divida la memoria lógica en trozos del mismo tamaño de cuadros, llamados páginas.

▪ Un programa con N páginas, necesita N marcos libres para ejecutarse.

▪ Mantenga una tabla de páginas y la dirección de marcos libres.

La tabla de páginas es la relación entre la página y donde está dentro de la memoria física

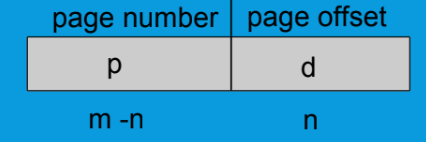
### **ESQUEMA DE TRADUCCIÓN DE DIRECCIONES**

▪ La dirección generada por la CPU se divide en:

▪ **Número de página** (p): se utiliza como índice en una tabla de páginas que contiene la dirección base de cada página en la memoria física

▪ **Desplazamiento de página** (d): combinado con la dirección base para definir la dirección de memoria física que se envía a la unidad de memoria

número de página desplazamiento de página

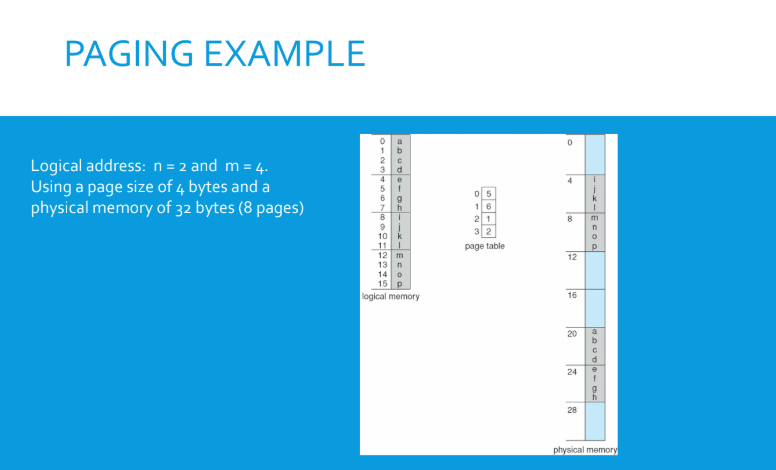


▪ Para un espacio de direcciones lógicas dado 2^m y tamaño de página 2^n

### TRADUCCIÓN DE PÁGINAS A MEMORIA FÍSICA

Dirección lógica: n = 2 ym = 4. Usando un tamaño de página de 4 bytes y una memoria física de 32 bytes (8 páginas)

### EJEMPLO DE PAGINADO



### PROBLEMAS CON LA PAGINA

▪ Requiere recursos adicionales del sistema operativo para mantener la tabla de páginas.

▪ No resuelve el problema de fragmentación interna.

▪ En el peor de los casos, un proceso requiere N páginas + 1 byte.

▪ En promedio, se desperdicia media página.

### PAGING - CÁLCULO DE LA FRAGMENTACIÓN INTERNA

▪ Tamaño de página = 2,048 bytes

▪ Tamaño del proceso = 72,766 bytes

▪ 35 páginas + 1,086 bytes

▪ Fragmentación interna de 2,048 - 1,086 = 962 bytes

▪ Fragmentación del peor caso = 1 fotograma - 1 byte

▪ En promedio fragmentación = 1/2 tamaño de cuadro

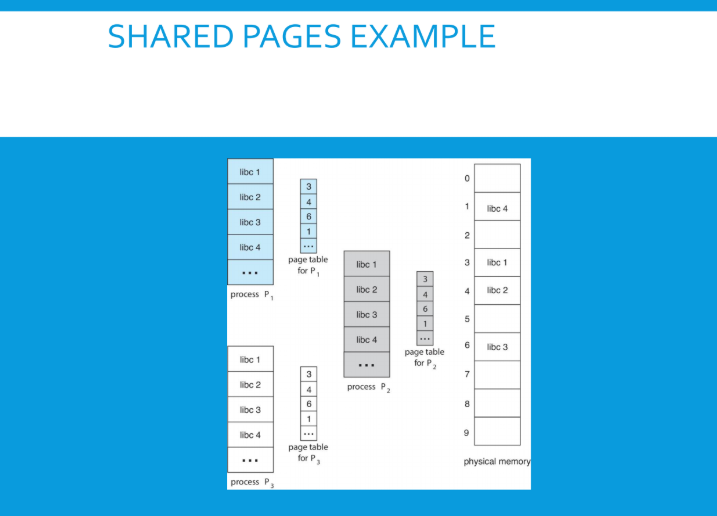
### BÚSQUEDA DE TABLA DE PÁGINAS

▪ Requiere dos accesos a la memoria, uno para la tabla de páginas y otro para las instrucciones / datos.

▪ Se puede resolver mediante el uso de memorias intermedias de traducción (TLB).

▪ No guarde toda la mesa, ya que puede acertar o fallar.

### EJEMPLO DE PÁGINAS COMPARTIDAS



### ESTRUCTURA DE LA TABLA DE PÁGINAS

▪ Las estructuras de memoria para paginación pueden volverse enormes usando métodos directos

▪ Considere un espacio de direcciones lógicas de 32 bits como en las computadoras modernas

▪ Tamaño de página de 4 KB (212)

▪ La tabla de páginas tendría 1 millón de entradas (232/212)

▪ Si cada entrada es de 4 bytes → cada proceso procesa 4 MB de espacio de direcciones físicas solo para la tabla de páginas

▪ No quiero asignar eso contiguamente en la memoria principal

▪ Una solución simple es dividir la tabla de páginas en unidades más pequeñas.

▪ Paginación jerárquica

▪ Tablas de páginas hash

▪ Tablas de páginas invertidas