U3 - Gestión de memoria

Introducción

La gestión de memoria es el proceso mediante el cual el sistema operativo organiza, controla y distribuye el espacio de memoria entre varios procesos que se están ejecutando. Su objetivo principal es garantizar que los procesos tengan suficiente memoria para funcionar de manera eficiente y que se puedan ejecutar tantos procesos como sea posible de manera simultánea. Podríamos decir que el gestor de memoria es un programa

Subdivisión de la memoria: El sistema operativo divide la memoria física en partes más pequeñas o "subdivisiones", permitiendo que varios procesos puedan residir en la memoria al mismo tiempo. Esto es fundamental para la **multiprogramación**, donde varios programas necesitan espacio en la memoria para ejecutarse.

Asignación eficiente: Es necesario asignar de forma eficiente la memoria disponible para maximizar el número de procesos que pueden ejecutarse al mismo tiempo. Esto requiere políticas de administración de memoria que minimicen el desperdicio de espacio, evitando **fragmentación interna** (donde el espacio dentro de las subdivisiones asignadas a los procesos no se usa por completo) y **fragmentación externa** (donde hay suficientes fragmentos de memoria libre, pero no contiguos).

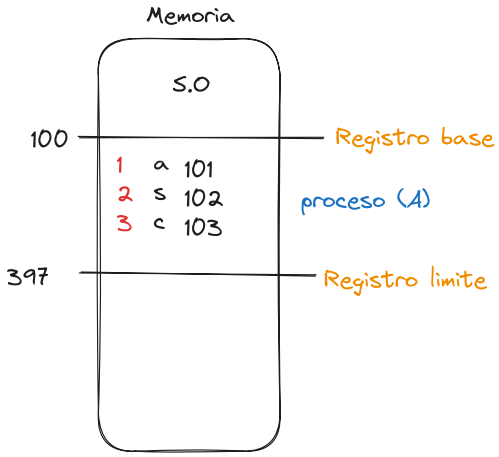
Direccionamiento de memoria

Lógica: es la dirección que me genera el compilador al por ejemplo crear una variable
Física: la física es la real de la memoria, es donde realmente esta el dato, y se suma la dirección lógica a la dirección del inicio del segmento (registro base)
Relativa: la dirección relativa es lo que me tengo que mover desde la dirección del inicio del segmento hasta la ubicación que esta dentro del segmento

> **Recordar**
>
> Para evitar que cada vez que busque algo del programa tiene que sumar la dirección lógica mas el registro base, lo que se hace es guardar en la posición relativa la suma una sola vez, entonces luego ya no tengo que sumarlo mas

Registro base y registro limite

El registro base es la guarda la dirección donde empieza el proceso o programa, y el registro limite guarda la dirección final.

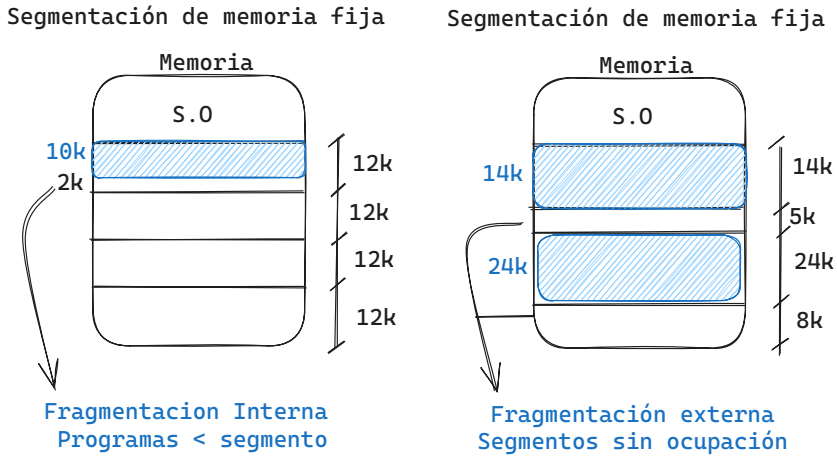


La dirección 1, 2, 3 en rojo marcan las direcciones relativas en la memoria y la columna en negro marca las física que es la suma de el registro base + las relativas, entonces cuando decimos de evitar la suma todo el tiempo guardamos de forma relativa la suma, y quedaría como la columna de direcciones en negro.

> **Importante**
>
> Las sumas de direcciones nunca pueden dar en valores menores al registro base y mayores al registro limite, sino estaría metiéndose en otro procesos

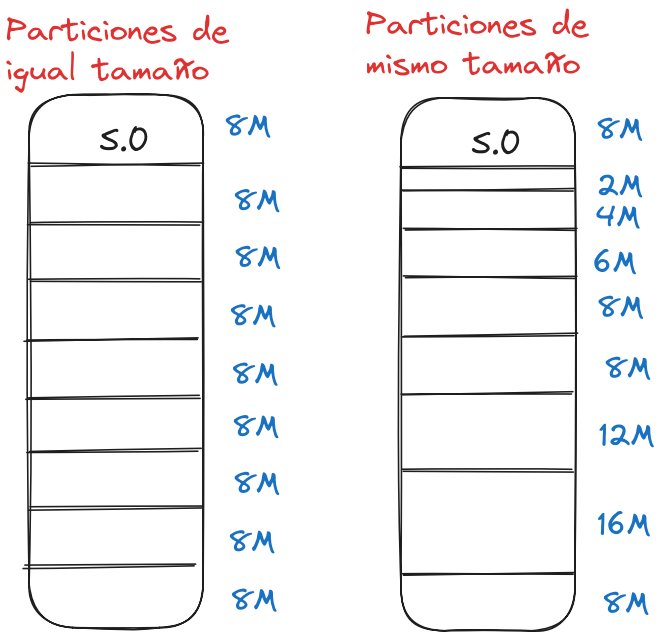
Reubicación de la memoria

Fraccionamiento interno y Fraccionamiento externo



- **Fraccionamiento interno**: Ocurre cuando se asigna más memoria de la necesaria a un proceso dentro de un bloque fijo, dejando espacio no utilizado dentro del bloque asignado.
- **Fraccionamiento externo**: Sucede cuando existen suficientes fragmentos de memoria libre, pero no contiguos, lo que impide asignar espacio a un nuevo proceso.

Diferentes particiones de memoria



Particiones de tamaño fijo igual: Es más simple de implementar pero tiende a generar mayor desperdicio de memoria (fragmentación interna) debido a la asignación rígida.

Particiones de tamaño fijo pero distintos: Proporciona mayor flexibilidad y reduce la fragmentación interna, pero introduce una mayor complejidad en la asignación de procesos y puede causar fragmentación externa.

A los algoritmos de ubicacion de particiones no darle muhca bola solo mencionarlos muy por encima

Técnicas de intercambio (Swapping)

El **Swapping** es una técnica de gestión de memoria que permite que un sistema operativo administre los recursos de manera más eficiente cuando hay una sobrecarga de procesos en la memoria física (RAM). El intercambio consiste en mover temporalmente procesos inactivos o de baja prioridad fuera de la memoria principal (RAM) hacia un espacio de almacenamiento secundario, típicamente el **disco duro**, y traerlos de vuelta cuando sea necesario.

Proceso de Swapping:

1. **Carga del Proceso**: Un proceso es cargado en la memoria RAM para ser ejecutado.
2. **Intercambio (Swap-Out)**: Si la RAM está llena o si el proceso no está en ejecución (inactivo), se mueve al almacenamiento secundario (espacio de swap).
3. **Liberación de Memoria**: La memoria ocupada por ese proceso es liberada para que otros procesos puedan utilizarla.
4. **Reactivación del Proceso (Swap-In)**: Cuando el proceso intercambiado necesita ser ejecutado nuevamente, se trae de vuelta a la RAM desde el disco.

Ventajas del Swapping:

- **Mayor flexibilidad**: Permite que más procesos "residan" en el sistema al mismo tiempo, aunque no todos estén en la memoria física simultáneamente.

- **Optimización del uso de la RAM**: Al intercambiar procesos que están inactivos o en espera, se libera memoria para procesos más activos o prioritarios.

Desventajas del Swapping:

- **Lentitud**: Mover procesos entre el disco y la RAM puede ser muy lento, ya que el acceso al disco es mucho más lento que el acceso a la memoria.
- **Fragmentación**: Puede generar fragmentación externa al liberar y reasignar bloques de memoria de diferentes tamaños.
- **Sobrecarga en el disco**: El espacio de swap puede saturarse si hay demasiados procesos en cola para ser intercambiados.

> **Recordar**
>
> Cuando vamos a cargar un proceso no hace falta que el proceso entero este cargado en la memoria, si solo cargo un parte y el resto en el disco puede ser posible. La parte que esta en memoria para iniciar el proceso se la llama **Conjunto residente**

Paginación

La **paginación** es una técnica de gestión de memoria que divide tanto la memoria física (RAM) como la memoria lógica (utilizada por los procesos) en bloques de tamaño fijo llamados **páginas** y **marcos**. La paginación permite que los procesos no tengan que ocupar un espacio contiguo en la memoria física, lo que facilita el manejo de la memoria y elimina los problemas de fragmentación externa que ocurren en otras técnicas como las particiones fijas.

Dirección virtual:

Es la palabra de la dirección de la memoria y se compone de dos partes:

- **Nro de pagina** Mas significativos
- **Desplazamiento** Menos significativos

Conceptos Clave:

1. **Página**: Es un bloque de memoria lógica (o virtual) de un proceso. Todos los procesos se dividen en páginas de tamaño uniforme.
2. **Marco**: Es un bloque de memoria física (RAM) en el que se almacena una página. Los marcos tienen el mismo tamaño que las páginas.
3. **Tabla de Páginas**: Cada proceso tiene una tabla de páginas que actúa como un "mapa" entre las páginas lógicas y los marcos de memoria física. Indica en qué marco está almacenada cada página de un proceso.

Tabla de pagina

La tabla de pagina es un conjunto de registros donde cada registro tiene unos bits reservados para referenciar la dirección de una pagina y el resto nos dan información al respecto de esa pagina, por

ejemplo

- **Presente/Ausente**: Indica si la pagina esta o no en la memoria RAM
- **Protected**: indica si se puede acceder a la pagina o no
- **Referenciada**: Indica si ha sido referenciada o utilizada la pagina
- **Modificada**: Indica si la pagina ha sido modificada o no
- **Desactivación de cache**: Indica inconsistencia en los datos

Funcionamiento:

- **División de procesos**: Cuando un proceso es cargado en memoria, se divide en páginas de tamaño fijo.
- **Asignación a marcos**: Estas páginas no necesitan estar cargadas en marcos contiguos de la memoria física. Cada página puede ser asignada a cualquier marco disponible.
- **Acceso a memoria**: Cuando el proceso necesita acceder a una página, la CPU consulta la tabla de páginas para encontrar en qué marco de la memoria física está almacenada esa página y accede a ella.

Este enfoque evita la **fragmentación externa**, ya que la memoria no se asigna en grandes bloques contiguos. Aunque puede haber **fragmentación interna** (si el proceso no utiliza completamente una página).

Ventajas de la paginación:

1. **Elimina la fragmentación externa**: Al permitir que las páginas y los marcos no tengan que ser contiguos en la memoria física, no se generan "espacios vacíos" entre procesos.
2. **Facilita el uso de la memoria virtual**: La paginación es la base de la memoria virtual, donde los procesos pueden exceder la capacidad física de la memoria usando el disco como una extensión.
3. **Memoria eficiente**: El tamaño de las páginas y marcos es fijo y pequeño, lo que mejora la utilización de la memoria.

Desventajas de la paginación:

1. **Fragmentación interna**: Si el tamaño de la página es mayor que los datos que necesita un proceso, el espacio restante dentro de la página se desperdicia.
2. **Sobrecarga de la tabla de páginas**: La necesidad de una tabla de páginas para cada proceso implica una sobrecarga en la memoria. En sistemas grandes, estas tablas pueden ser bastante grandes.
3. **Tiempos de acceso más lentos**: Consultar la tabla de páginas en cada acceso a la memoria puede generar tiempos de acceso ligeramente más largos. Sin embargo, esto se puede mitigar usando una **TLB** (Translation Lookaside Buffer), que almacena en caché las traducciones de páginas recientes.

Segmentación

MMU (Memory Managment Unit)

La **Unidad de Gestión de Memoria** o **MMU** (Memory Management Unit) es un componente hardware responsable de manejar la traducción de direcciones virtuales a direcciones físicas en los sistemas operativos modernos. Actúa como intermediario entre la CPU y la memoria, permitiendo que los programas utilicen direcciones virtuales en lugar de las direcciones físicas reales de la memoria. Esto es fundamental para la implementación de características como la **memoria virtual**, **protección de memoria**, y la **paginación**.

Funciones principales de la MMU:

1. **Traducción de direcciones**:
   - Cuando un programa accede a una dirección de memoria, está utilizando una dirección **virtual**. La MMU convierte esta dirección virtual en una **dirección física** que apunta a un lugar específico en la memoria RAM.
   - Esta traducción es esencial porque cada proceso tiene su propio espacio de direcciones virtuales, lo que permite que los programas se ejecuten sin interferir entre sí.
2. **Paginación**:
   - La MMU gestiona la paginación, un mecanismo que divide la memoria en pequeñas secciones llamadas **páginas** (para la memoria virtual) y **marcos** (para la memoria física). Con esto, los procesos no necesitan ocupar áreas contiguas de la memoria física.
   - Cuando una página de un proceso no está presente en la memoria física, la MMU puede generar una **falla de página** (page fault), que es gestionada por el sistema operativo para cargar la página desde el disco duro.
3. **Protección de memoria**:
   - La MMU proporciona mecanismos para evitar que los procesos accedan a la memoria asignada a otros procesos.
   - Establece permisos de acceso (lectura, escritura, ejecución) a diferentes regiones de la memoria, lo que ayuda a mantener la seguridad y estabilidad del sistema.
4. **Soporte para memoria virtual**:
   - La MMU es clave para la implementación de la **memoria virtual**, un mecanismo que permite que un sistema utilice más memoria de la que físicamente tiene disponible, utilizando el disco duro como una extensión de la RAM.
   - Cuando un proceso necesita más memoria de la que tiene disponible, la MMU permite mover páginas de la memoria a la **memoria secundaria** (como el disco duro) y viceversa.

Esquema de funcionamiento básico de la MMU:

1. El programa intenta acceder a una dirección de memoria virtual.
2. La CPU envía la dirección virtual a la MMU.
3. La MMU busca en su tabla de páginas la dirección física correspondiente a esa dirección virtual.
4. Si la página está en la RAM (memoria física), la MMU envía la dirección física a la CPU, y el acceso a la memoria continúa.
5. Si la página no está en la RAM, se produce una **falla de página**, y el sistema operativo debe cargar la página desde el disco a la memoria antes de continuar.

Beneficios de la MMU:

- **Abstracción de la memoria**: Permite que los programas utilicen direcciones virtuales y no se preocupen por la ubicación física de los datos.
- **Aislamiento de procesos**: Cada proceso tiene su propio espacio de direcciones, lo que impide que interfieran entre sí.
- **Seguridad**: Ayuda a prevenir el acceso no autorizado a áreas de memoria asignadas a otros procesos o al sistema operativo.
- **Optimización del uso de memoria**: Gracias a la paginación y la memoria virtual, se puede optimizar el uso de la memoria física y permitir que el sistema soporte más procesos simultáneamente.

En resumen, la **MMU** es una parte esencial del hardware que permite la traducción de direcciones y la gestión eficiente de la memoria en sistemas modernos, garantizando la seguridad, el uso óptimo de la memoria, y permitiendo la ejecución de múltiples procesos de manera aislada y segura.

> **Hiperpaginacion o trashing**
>
> Es cuando tengo recursos exasos hablando de software, y ejecuto pocos programas y se me llena el swap, pero podria pensar en recalcular el swap ponerlo mas grande y no tendria problemas y podria correr mas programas, pero esto es ineficiente ya que no tendria casi nada en memoria y tendria casi todo en el swap, y eso le llearia mas tiempo al procesador de andar leyendo todo desde el swap que lo que estaria ejecutando, por lo tanto es mejor evitarlo y eso se puede evitar elevando la memoria RAM por ejemplo.

APUNTES DE CLASE

TLB (Buffer de dirección adelantada)

Es como un grupo de registros (no se sabe cuantos) que lo que hace es que cuadno se busca una direccion de memoria en las tablas de pagina, hay un bit que no sindica si se encontro en la tabla de pagina o no, es decir **acierto de pagina** y si lo encuentra manda esa direccion a la TLB, ya que probablemente los siguientes direcciones esten muy cer

Tiene los mismo datos que tiene la tabla de pagina, Esto hace que funcione la segmentación mas rápido

BitMap (Mapa de bits)

Es un registro que tiene tantos bits como marcos de pagina de la memoria, que tienen 1 (ocupado) o `0` (vacio) en base a si esta ocupado o no el marco de pagina, para saber si esta libre la ram o no.

> **La diferencia entre swap y disco**
>
> Si tengo que mandar una parte de un proceso (pagina) al swap o al disco porque se me lleno la ram, la diferencia entre mandarlo al swap o al disco se basa en que si es solo codigo lo envio al disco ya que no cambio y sigue igual, pero si son datos y se han modificado es mejor dejarlo en el swap.