

Tema 2

① Si tuvieras que quitar algo sería lo más difícil de el código para realizarlo no, porque no tiene la posibilidad de hacerlo ya que no se sabe de qué es lo que puedes eliminar frente al espacio que liberas.

②

Se informa de que se ha producido un error ya que al acceder a una dirección de memoria fuera del área asignada puede escribir en otra celda de memoria que no corresponde al proceso y provocar un fallo en el sistema.

Informa el S.O. concretamente el gestor de memoria que lanza una excepción.

③

Si tiene sentido, un solo usuario puede tener acabado la multiprogramación, es decir permitir que se ejecuten simultáneamente más de un proceso, cuyos datos e instrucciones se encuentran en memoria principal.

Ambos conceptos no son incompatibles

Y en un entorno de multiprogramación, el diagrama de estados es fundamental, para que el S.O. determine los puntos de intercambio y asignación de recursos a cada proceso.

④

c) FINALIZADO

Por ejemplo si a un proceso se le asigna una dirección de memoria que no existe, se finalizará (si se bloquea no es posible reanudarlo)

El SO abre la ejecución de este programa.

Si se bloquea es porque en algún momento se dan las circunstancias necesarias para su reanudación.

Por ejemplo cuando redireccionar algo y no entra en la memoria principal el proceso se bloquea hasta que llegue el espacio de memoria que falta.

⑤

No tiene sentido, porque solo puede tener una tarea, es decir un proceso ejecutándose en memoria, ejecutándose luego un planificador de procesos no es coherente si solo sirve a ejecutar un programa a la vez y no se execute otro tarea que termine el anterior.

⑥

Bajo ninguna circunstancia ya que el planificador de procesos es la parte del sistema operativo que se encarga de decidir qué proceso emplea el procesador en cada instante, por tanto es el encargado de implementar la multiprogramación.

Sin el planificador de procesos no serviría capacer de gestionar los recursos disponibles entre los distintos procesos que se están ejecutando de forma simultánea ni de repartir correctamente el tiempo de CPU entre los distintos procesos.

Interrupciones pueden
ser en cualquier momento

⑦

- b) Cambiar la fecha del sistema.
- c) Leer una pista / sector de un disco magnético.
- d) Modificar la dirección de un vector de la tabla de vectores de interrupción
- e) Deshabilitar interrupciones

⑧

Es necesario proteger al sistema operativo y a los estructuras de datos importantes, tales como los bloques de control de procesor, de las instrucciones de los programas de usuario.

Si se produce algún tipo de error en modo usuario el supervisor es un usuario privilegiado capaz de acceder al sistema en un modo protegido, es decir el programa (S0) tiene el control completo del procesador y de todos los instrucciones, registros y memoria.

⑨

a) Es necesario, para que el SO libere memoria y pueda ejecutar otro programa, en un monoprogramado también tiene sentido (es decir en un monoprogramado) se ejecuta un programa y hasta que no termina no comienza otro, luego con más razón es necesario que se realice la liberación al sistema.

b) No necesariamente.

Por ejemplo estás ejecutando un proceso de usuario y se llega a una instrucción que muestra una operación en tal caso abre un fichero. Esta liberación provoca

le transferencia o una memoria
P. Ejemplo: (cuando no siempre) el uso de una memoria (llamada al
sistema hace que el proceso de un solo paso a otro sea
bloqueado. (c?)

- c) No, cambiar el valor del estado de un proceso
solamente se puede hacer en modo kernel o supervisor
- d) Si, dejando paso a otro proceso que estaba en
espera y que pasa a ejecutarse.
- e) No, el proceso se ejecuta cuando el planificador lo
determine (no pueden cambiar las prioridades)
pero no los estados

10

Cuando las operaciones de E/S son manejadas
directamente por la CPU, es decir no se dispone
ni de interrupciones ni de DMA, se requiere la
intervención activa del procesador para transferir
los datos entre la memoria y el módulo de E/S
lo que en definitiva implica que el procesador
no dispondrá de mucho tiempo para la ejecución
de procesos en medio de las operaciones de E/S
de un proceso. En cambio cuando dispone
de DMA, el procesador sólo se ve involucrado al
principio y al final de la transferencia,
estando desocupado en el transcurso de la operación de E/S
y pudiendo aprovechar este tiempo para ejecutar
otro proceso (multiprogramación).

14

Es probable que si forma parte del sistema operativo sera más difícil de depurar, sería más complejo, menos fiable, ocuparía más memoria y no es eficaz.

Además se podrían ejecutar órdenes en modo kernel / supervisor lo cual puede llegar a ser muy peligroso, ya que estos cambios se realizan a través de llamadas al sistema después de evaluar si la orden es segura o no.

12

- a) Depende de si se ejecuta o no "en decir depende" de la prioridad y la posición que ocupe en la cola"
- b) Seguramente si, ya que dejará de estar ejecutándose y pasará a ejecutarse otro proceso.
- c) No, necesariamente, depende si se va a ejecutar otro proceso
- d) Si se encuentra en la primera posición de la cola si, sino no.
- e) No implica un cambio de contexto, a menos que sea el primero de ellos y pase a ejecutarse

13

Tiene sentido en el caso por ejemplo de que dos procesos estén bloqueados por la misma razón, cuando estos pasen a prepararse y se rayan a ejecutar, se utilizan la prioridad

14

Como los direcciones están expresadas en binario, es más fácil separar el segmento del desplazamiento. El enigma de dirección lógica es transparente al programador, al ensamblador y al montador. Cada dirección lógica de un ~~programador~~ programa es idéntica a la real.

15

a) 10 de desplazamiento \Rightarrow el tamaño de cada página "son" las posibles combinaciones con 10 "bytes". Luego $\frac{2^{20}}{2^{10}} = 1 \text{ KByte}$. Imaginemos que 1 despl. es 4 bytes.

b) Se divide la memoria real (16MBytes) por el tamaño de cada página.

$$\frac{2^{20} \cdot 2^4}{2^{10}} \Rightarrow \text{memoria física en Bytes} = 2^{14} \text{ mbytes}$$

c) 14 bits

d) 103KB + memoria lógica.

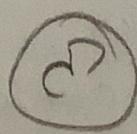
$$\frac{103 \text{ KB}}{1 \text{ KB por pag}} = 103 \text{ páginas. Luego}$$

103 filas de la tabla.

$$103 \cdot (14+1) = 1545$$

↓ ↓

bit de protección



16

a) 999

$$\frac{1024}{999} = \frac{999}{1024 \text{ bytes}} = 0 \text{ resto } 999$$

por que 0 => estoy en la 4 física

DIRECCIÓN FÍSICA = NÚMERO DE MARCO • TAMAÑO PÁGINA + DESPLAZAMIENTO

$$DF = 4 \cdot 1024 + 999 = 5095$$

b)

$$\frac{2121}{1024} = 2 \text{ resto } 73.$$

$$DF = 1 \cdot 1024 + 73 = 1097$$

c)

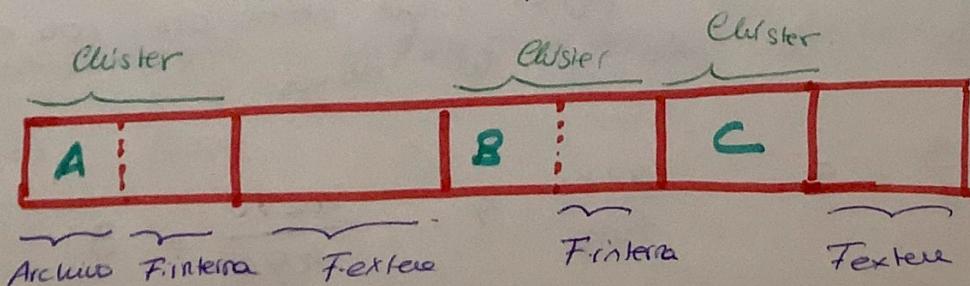
$$\frac{5400}{1024} = 5 \text{ resto } 280$$

$$DF = 0 \cdot 1024 + 280 = 280$$

17

Buscar en internet

En el método de paginación se produce fragmentación interna. Es decir (pérdida de espacio en disco) debido al hecho de que el tamaño de un determinado cluster sea inferior al tamaño del clúster, en el caso de la fragmentación externa aparece como consecuencia de las distintas políticas de ajuste de bloques que tiene un sistema de ficheros.



En este caso como hemos dicho cuantos fragmentos
internos (los páginas) no se completan al 100%)

de forma de solucionar este problema sería hacer
páginas más pequeñas y por tanto más
versátiles. Sin embargo esto contribuiría al deterioro
del proceso.

(18)

$$\text{Diracción lógica} = 2453$$

Técnica de paginación con páginas de 1024

$$\frac{2453}{1024} \Rightarrow \text{cociente } 2, \text{ resto } 405 \quad (\text{mrgo } (2,405))$$

No es posible traducirla a la dirección física
ya que

$$9322 = x \cdot 1024 + 405 \Rightarrow 9322 - 405 = 1024x \Rightarrow x = 8'7$$

Mrgo 8'7 C/M.

(19)

Paginación

$$\text{Memoria física} = 131072 \text{ bytes}$$

a) Supongamos el primer caso, 4096 bytes. NO CORRECTA

Sabiendo que una página no puede contener parte
de dos segmentos diferentes, debemos calcular
por separado las páginas que necesitan para el
código, la pila y los datos.

$$\frac{20480}{4096} = 5 \quad \frac{14288}{4096} = 4 \quad \frac{10240}{4096} = 3 \quad \left\{ 12 \right.$$

B

$$\frac{16384}{4096} = 4 \quad \frac{8200}{4096} = 3 \quad \frac{8192}{4096} = 2 \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} 9$$

C

$$\frac{18432}{4096} = 5 \quad \frac{13288}{4096} = 4 \quad \frac{9216}{4096} = 3 \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} 12$$

En total 33 páginas

$33 \text{ páginas} \times 4096 \text{ bytes} = 135168 > \text{Memoria de lo que disponemos}$
luego no es posible

Otra forma de comprobarlo es $\frac{131072 \text{ bytes}}{4096 \text{ bytes}} =$
~~32~~ 32

Es decir tener disponer 32 páginas < 33
luego esta operación no es posible.

Supongamos el segundo caso, 512 bytes ← CORRECTA

A

$$\frac{20480}{512} = 40 \quad \frac{14288}{512} = 28 \quad \frac{10240}{512} = 20 \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} 88$$

B

$$\frac{16384}{512} = 32 \quad \frac{8200}{512} = 17 \quad \frac{8192}{512} = 16 \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} 65$$

C

$$\frac{18432}{512} = 36 \quad \frac{13288}{512} = 26 \quad \frac{9216}{512} = 18 \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} 80$$

En total 833 páginas

$833 \text{ páginas} \times 512 \text{ bytes} = 119296 < \text{Memoria de lo que disponemos}$
o de lo que tiene.

$\frac{131072}{512} = 256 \text{ páginas}$ de los que disponemos, $256 > 233$

Luego esta es la operación correcta

b) Necesitamos

→ 1 bit de protección

→ Tamaño de la página 2^9 19 bits

→ Dirección del marco $\Rightarrow \frac{31072B}{512B} = 256$ marcos $\Rightarrow 2^8$ 8 bits

$$1 + 9 + 8 = \underline{\underline{18 \text{ bits}}}$$

c) Habrá tres tablas de páginas (una por proceso)

Necesitaremos tantos entradas como páginas hayo en este table de páginas

A ~ 88 entradas

B ~ 61 entradas

C ~ 80 entradas

②0 dar tabla de páginas

✗

②1

Primer proceso: $\frac{31566B}{2048B} = 15$ págs completas resto 846 + 1 fragmentación.

Segundo proceso: $\frac{18432B}{2048B} = 9$ páginas completas.

En el primer proceso $\Rightarrow 2048 - 846 = 1202B$ fragmento.

En el segundo no ug.

22

segmentación

- comprobar que el desplazamiento es menor que la longitud
(en el apartado d) NO)

seg 0

• a) $430 < 600 \Rightarrow DF = \underline{\underline{BASE + DESP}} \Rightarrow 219 + 430 = 649$

seg 1

• b) $10 < 14 \Rightarrow DF = \underline{\underline{BASE + DESP}} \Rightarrow 2300 + 10 = 2310$

seg 3

• c) $400 < 500 \Rightarrow DF = \underline{\underline{BASE + DESP}} \Rightarrow 1527 + 400 = 1727$

23

Es más rápido cambiar de un WLS a otro dentro del mismo proceso, que cambiar de un proceso a otro. En la forma se debe a que los WLS comparten datos y espacios de direcciones, mientras que los procesos al ser independientes no lo hacen.