SO-Relacion-Tema-3.pdf

Frankie2

Sistemas Operativos

2º Grado en Ingeniería Informática

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
Universidad de Granada

WUOLAH

El más PRO del lugar puedes ser Tú.

¿Quieres eliminar toda la publi de tus apuntes?

¡Fuera Publi!
Concéntrate al máximo

Apuntes a full.
Sin publi y sin gastar coins

Para los amantes de la inmediatez, para los que no desperdician ni un solo segundo de su tiempo o para los que dejan todo para el último día.

Quiero ser PRO

4,95 / mes

1. **Si un computador no posee hardware de reubicación, e implementa intercambio (swapping), entonces el gestor de memoria necesita utilizar un cargador para recalcular las direcciones físicas de un proceso. ¿Sería posible para el sistema de intercambio reubicar los segmentos de datos y pila? Explica cómo funcionaría este sistema, o si es imposible que funcione.**
   Sería imposible que funcionase, ya que por ejemplo, si hay direcciones físicas que sean punteros, el cargador traduciría la dirección del puntero pero no a dónde apunta el puntero. El sistema operativo no sabe cómo traducir estos punteros.

2. **Considere un sistema con un espacio lógico de memoria de 128K páginas (máximo espacio de memoria virtual) con 8 KB cada una, una memoria física de 64 MB y direccionamiento al nivel de byte. ¿Cuántos bits hay en la dirección lógica? ¿Y en la física?**
   Espacio de direcciones físicas = 64 MB = 2^{26} B → 26 bits de dirección física
   Páginas = 128K = 2^{17} → 17 bits para identificar todas las páginas
   Tamaño de página = 8 KB = 2^{13} B → 13 bits del campo de desplazamiento

   Dirección física 26 bits → 13 bits de desplazamiento + 13 bits de marco
   Dirección lógica 30 bits → 13 bits de desplazamiento + 17 bits de página

3. **Considérese un sistema con memoria virtual en el que el procesador tiene una tasa de utilización del 15% y el dispositivo de paginación está ocupado el 97% del tiempo, ¿qué indican estas medidas? ¿Y si con el mismo porcentaje de uso del procesador el porcentaje de uso del dispositivo de paginación fuera del 15%?**
   Si el dispositivo de paginación está ocupado al 97%, significa que se está intentando acceder a bloques de memoria que no están cargados en memoria principal, y por tanto el mecanismo de paginación tiene que estar trabajando constantemente para traducir direcciones virtuales a direcciones físicas. Esto quizás se debe a que la memoria virtual es escasa o a una mala política de reemplazo. Este problema se conoce como *hiperpaginación*.

   Si el porcentaje de uso del dispositivo de paginación fuera del 15%, significa que gran parte de los bloques de memoria que se *piden* están cargados en memoria principal. Esto quizás se debe a un tamaño aceptable de la memoria virtual o al uso de una buena política de reemplazo. Sería necesario aumentar el grado de multiprogramación llamando al procesador a corto plazo, utilizando swapping...

4. **Sea un sistema de memoria virtual paginada con direcciones lógicas de 32 bits que proporciona un espacio virtual de 2^{20} páginas y con una memoria física de 32 Mbytes ¿cuánta memoria requiere en total un proceso que tenga 453 Kbytes, incluida su tabla de páginas cuyas entradas son de 32 bits?**
   Sabemos que tenemos direcciones lógicas de 32 bits. De esos 32 bits, 20 bits son para el número de página (el enunciado nos dice que tenemos 2^{20} páginas) y el resto de bits serán para el desplazamiento.

   De los bits de desplazamiento podemos obtener el tamaño de página:
   Tamaño de página = 2^d = 2^{12} B/página

   Así, sabiendo cuánto ocupa el proceso, podemos obtener las páginas que necesita el proceso:

   453 KB * 1024 B = 463.872 B
   463.872 B / 2^{12} B/página = 114 páginas necesita nuestro proceso

   Lo que el proceso ocupa en memoria es lo siguiente: lo que ocupan sus páginas más lo que ocupa su tabla de páginas

   Sus páginas, 114 ocupan = 114 * 2^{12} B/página = 114 páginas * 4096 B/página = 466944 B

   Su tabla de páginas ocupará 114 páginas por 32 bits (4B) que ocupa cada entrada de página.

   Su tabla de páginas ocupa = 114 páginas * 4B/página = 456 B

   Por tanto, el tamaño total del proceso en memoria es de 467.400 B

5. **Un ordenador tiene 4 marcos de página. En la siguiente tabla se muestran: el tiempo de carga, el tiempo del último acceso y los bits R y M para cada página (los tiempos están en tics de reloj). Responda a las siguientes cuestiones justificando su respuesta.**
   a. **¿Qué página se sustituye si se usa el algoritmo FIFO?**
      Se sustituye la página 2, ya que es la primera que entró según el tiempo de carga.

   b. **¿Qué página se sustituye si se usa el algoritmo LRU?**
      Se sustituye la página 3, ya que es la que tiene la referencia más antigua.

6. **¿Depende el tamaño del conjunto de trabajo de un proceso directamente del tamaño del programa ejecutable asociado a él? Justifique su respuesta.**
   No tiene por qué. El conjunto de trabajo de un programa está formado por las páginas que se referencian en un periodo determinado de tiempo. Que un programa sea de gran tamaño no implica que en ese intervalo de tiempo referencie a muchas páginas, bien puede referenciar a muchas o a dos o tres.

7. **¿Por qué una caché (o la TLB) que se accede con direcciones virtuales puede producir incoherencias y requiere que el sistema operativo la invalide en cada cambio de contexto y, en cambio, una que se accede con direcciones físicas no lo requiere?**
   Esto se debe a que las direcciones virtuales son traducidas dinámicamente en direcciones físicas en tiempo de ejecución. Cuando se produce un cambio de contexto, estas direcciones pueden variar, por tanto, es necesario invalidarlas.

8. **Un ordenador proporciona un espacio de direccionamiento lógico (virtual) a cada proceso de 65.536 bytes de espacio dividido en páginas de 4096 bytes. Cierto programa tiene un tamaño de región de texto de 32768 bytes, un tamaño de región de datos de 16386 bytes y tamaño de región de pila de 15878 bytes. ¿Cabría este programa en el espacio de direcciones? (Una página no puede ser utilizada por regiones distintas). Si no es así, ¿cómo podríamos conseguirlo, dentro del esquema de paginación?**
   Primeramente obtenemos el número de páginas que tenemos disponibles:

   65536 bytes / 4096 bytes/página = 16 páginas disponibles en nuestro ordenador

   Ahora vemos cuántas páginas ocupa cada región de nuestro programa:

   Región de texto: 32768 bytes / 4096 bytes/página = 8 páginas
   Región de datos: 16386 bytes / 4096 bytes/página = 5 páginas
   Región de pila: 15878 bytes / 4096 bytes/página = 4 páginas

   El programa requiere un total de 17 páginas, por tanto, no cabría el programa en el espacio de direcciones del ordenador. Presenta un problema de fragmentación interna.

   La solución al problema consiste en disminuir lo suficiente el tamaño de página hasta que el programa cupiese en el espacio de direcciones.

9. **Analice qué puede ocurrir en un sistema que usa paginación por demanda si se recompila un programa mientras se está ejecutando. Proponga soluciones a los problemas que pueden surgir en esta situación.**
   Probablemente ocurrirán muchos fallos de página, ya que el sistema habría traído una serie de páginas contiguas en el instante de ejecución que probablemente no sean referenciadas al ser recompilado el programa.

   Una solución podría ser el esperar a que el programa esté compilado para de nuevo aplicar paginación por demanda e ignorar las páginas que hemos traído a memoria en el instante de recompilación.

10. **Para cada uno de los siguientes campos de la tabla de páginas, se debe explicar si es la MMU o el sistema operativo quién los lee y escribe (en éste último caso si se activa o desactiva), y en qué momentos:**
    a. **Número de marco**
    Es escrito por el Sistema Operativo y leído por el MMU a la hora de traducir una dirección lógica.

    b. **Bit de presencia**
    El MMU lo lee a la hora de traducir una dirección lógica, para saber si se encuentra en memoria principal. El MMU es quien genera la excepción en caso de que el bit sea 0. El Sistema Operativo lo modifica (al crear una página y cuando resuelve una falta de página).

    c. **Bit de protección**
    Lo modifica el SO al crear la tabla de páginas del proceso para indicar si se puede acceder a la página o no y lo lee el MMU a la hora de traducir de dirección lógica a dirección física. (Rutina de excepción de falta de página).

    d. **Bit de modificación**
    Lo activa el MMU al hacer la traducción y el SO lo desactiva cuando carga la página en memoria principal. El SO también lo lee cuando utiliza un algoritmo de sustitución de página.

    e. **Bit de referencia**
    Lo modifica el SO cuando es necesario indicar si una página ha sido referencia o no. Lo activa el MMU cuando activa una página y también el SO en el algoritmo de reloj.

11. **Suponga que la tabla de páginas para el proceso actual se parece a la de la figura. Todos los números son decimales, la numeración comienza en todos los casos desde cero, y todas las direcciones de memoria son direcciones en bytes. El tamaño de página es de 1024 bytes.**

Número de página virtual	Bit de validez o presencia	Bit de referencia	Bit de modificación	Número de marco de página
0	0	1	0	4
1	1	1	1	7
2	1	0	0	1
3	1	0	0	2
4	0	0	0	-
5	1	0	1	0

¿Qué direcciones físicas, si existen, corresponderán con cada una de las siguientes direcciones virtuales? (no intente manejar ninguna falta de página, si las hubiese)

a. **999**
   999/1024 = 0,... Se corresponde al número de página virtual 0, que tiene el bit de presencia a 0, por tanto la página no está presente en memoria.

b. **2121**
   2121/1024 = 2,... Se corresponde al número de página virtual 2, que se encuentra en el marco de página 1. Por tanto, la dirección física es:

   El resto de la división anterior (desplazamiento): 2121-(1024*2) = 73

   (1024*1) + 73 = 1097

c. **5400**
   5400/1024 = 5,... Se corresponde al número de página virtual 5, que se encuentra en el marco de página 0. Por tanto, la dirección física es:
   El resto de la división anterior (desplazamiento): 5400-(1024*5) = 280
   (1024*0) + 280 = 280

12. **Sea la siguiente secuencia de números de página referenciados: 1,2,3,4,1,2,5,1,2,3,4,5. Calcula el número de faltas de página que se producen utilizando el algoritmo FIFO y considerando que el número de marcos de página de que disfruta nuestro proceso es de:**

a. **3 marcos**
   Secuencia:
   1 2 3 4 1 2 5 1 2 3 4 5

   Tiempo:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1	1	1	4	4	4	5	5	5	5	5	5	
	2	2	2	1	1	1	1	1	3	3	3	
		3	3	3	2	2	2	2	2	4	4	
*	*	*	*	*	*	*			*	*		

   Se producen 9 faltas de página

b. **4 marcos**

Secuencia:

1 2 3 4 1 2 5 1 2 3 4 5

Tiempo:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1	1	1	1	1	1	5	5	5	5	4	4	
	2	2	2	2	2	2	1	1	1	1	5	
		3	3	3	3	3	3	2	2	2	2	
			4	4	4	4	4	4	3	3	3	
*	*	*	*			*	*	*	*	*	*	

Se producen 10 faltas de página

¿Se corresponde esto con el comportamiento intuitivo de que disminuirá el número de faltas de página al aumentar el tamaño de memoria de que disfruta el proceso?

Claramente no se corresponde, ya que hemos obtenido más faltas de página usando un marco más.

13. **¿Por qué la localidad no es un factor que se tiene en cuenta en los sistemas con segmentación?**

    Porque al ser los segmentos de tamaño variable, toda la información necesaria del segmento está incluida en el mismo.

14. **En la gestión de memoria en un sistema paginado, ¿qué estructura/s de datos necesitará mantener el Sistema Operativo para administrar el espacio libre?**

    La tabla de marcos de página, que contiene información sobre cada marco de página, la tabla de ubicación en disco, que mantiene la ubicación de cada página en el almacenamiento secundario y la tabla de páginas donde se almacena toda la información necesaria para poder realizar la traducción de direcciones lógicas a direcciones físicas.

15. **¿Cuánto puede avanzar como máximo la aguja del algoritmo de reemplazo de páginas del reloj durante la selección de una página?**

    Como máximo, podrá avanzar el mismo número de pasos como marcos de página haya, ya que, suponiendo que el bit de referencia de todos los marcos consultados sea R=1, tendrá que volver a llegar al inicio ya que será el primer marco que puso a R=0 el que sustituirá para seleccionar una página.

16. Situándonos en un sistema paginado, donde cada proceso tiene asignado un número fijo de marcos de páginas. Supongamos la siguiente situación: existe un proceso con 7 páginas y tiene asignados 5 marcos de página. Indica el contenido de la memoria después de cada referencia a una página si como algoritmo de sustitución de página utilizamos el LRU (la página no referenciada hace más tiempo). La secuencia de referencias es la indicada en la figura.

Secuencia:

2 1 3 4 1 5 6 4 5 7 4 2

Tiempo:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
2	2	2	2	2	2	6	6	6	6	6	6	
	1	1	1	1	1	1	1	1	1	1	2	
		3	3	3	3	3	3	3	7	7	7	
			4	4	4	4	4	4	4	4	4	
					5	5	5	5	5	5	5	
*	*	*	*		*	*			*		*	

Se producen 8 faltas de página

17. ¿Cuál es la ventaja del algoritmo de frecuencia de faltas de página sobre el algoritmo basado en el modelo del conjunto de trabajo utilizando el tamaño de ventana w? ¿Cuál es la desventaja?

El de faltas de página es menos costoso porque se activa solo cuando hay una falta de página, el otro es más fiable pero se activa en cada referencia. La desventaja es que si FPP se traslada a una nueva región de referencias, se producen muchas faltas de página.

18. Supongamos que tenemos un proceso ejecutándose en un sistema paginado, con gestión de memoria basada en el algoritmo de sustitución frecuencia de faltas de página. El proceso tiene 5 páginas (0, 1, 2, 3, 4). Represente el contenido de la memoria real para ese proceso (es decir, indique qué páginas tiene cargadas en cada momento) y cuándo se produce una falta de página. Suponga que, inicialmente, está cargada la página 2, el resto de páginas están en memoria secundaria y que no hay restricciones en cuanto al número de marcos de página disponibles. La cadena de referencias a página es: 0 3 1 1 1 3 4 4 2 2 4 0 0 0 0 3 y el parámetro es τ=3.

Secuencia:

0	3	1	1	1	3	4	4	2	2	4	0	0	0	0	3

Tiempo:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	2	2	2	2	2	2	-	-	2	2	2	2	2	2	2	2
	0	0	0	0	0	0	-	-	-	-	-	0	0	0	0	0
		3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
			1	1	1	1	1	1	1	1	1	1	1	1	1	1
							4	4	4	4	4	4	4	4	4	4
	*	*	*				*		*			*				

Se producen 6 faltas de página

19. Describa el funcionamiento del algoritmo de sustitución basado en la frecuencia de faltas de página, con los siguientes datos: 4 marcos de página, en t = 0 la memoria contiene a la página 2. El tamaño de la ventana es τ= 3 y se produce la secuencia de referencias de páginas, 1 4 2 2 2 4 5 5 3 3 5 1 1 1 1 4.

Secuencia:

1	4	2	2	2	4	5	5	3	3	5	1	1	1	1	4

Tiempo:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	2	2	2	2	2	2	2	2	2	2	2	2				
	1	1	1	1	1	1	5	5	5	5	5	5				
		4	4	4	4	4	4	4	4	4	4	4				
									3	3	3	3				
	*	*					*		*			*!				

Se producen 5 faltas de página. En el instante t=12, además, el proceso se bloquea porque no tenemos marcos de página disponibles. Deberá esperar a tener los recursos necesarios para ampliar su tamaño.

20. Describa el funcionamiento del algoritmo de sustitución global basado en el algoritmo basado en el modelo del conjunto de trabajo, con los siguientes datos: 4 marcos de página, en t= 0 la memoria contiene a la página 2 que se referenció en dicho instante de tiempo. El tamaño de la ventana es τ= 3 y se produce la secuencia de referencias de páginas, 1 4 4 4 2 4 1 1 3 3 5 5 5 5 1 4.
Secuencia:

1	4	4	4	2	4	1	1	3	3	5	5	5	5	1	4

Tiempo:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	2	2	-	-	2	2	2	-	3	3	3	3	5	5	5	5
	1	1	1	-	-	-	1	1	1	1	5	5			1	1
		4	4	4	4	4	4	4	-							4
	*	*			*		*		*		*				*	*

Se producen 8 faltas de página

21. Una computadora con memoria virtual paginada tiene un bit U por página virtual, que se pone automáticamente a 1 cuando se realiza un acceso a la página. Existe una instrucción limpiar_U (dir_base_tabla) que permite poner a 0 el conjunto de los bits U de todas las páginas de la tabla de páginas cuya dirección de comienzo pasamos como argumento. Explica cómo puede utilizarse este mecanismo para la implementación de un algoritmo de sustitución basado en el modelo del conjunto de trabajo.
Este mecanismo se podría utilizar para dereferenciar todas las páginas de la tabla que estén fuera del intervalo (t-x, t] del algoritmo basado en el modelo del conjunto de trabajo.

22. Un Sistema Operativo con memoria virtual paginada tiene el mecanismo fijar_página(np) cuyo efecto es proteger contra la sustitución al marco de página en que se ubica la página virtual np. El mecanismo des_fijar (np) suprime esta protección.

a. **¿Qué estructura/s de datos son necesarias para la realización de estos mecanismos?**
Es necesario tener un bit por cada entrada en la tabla de páginas para indicar si está fijada o no.

b. **¿En qué caso puede ser de utilidad estas primitivas?**
Sería útil para búferes de entrada/salida, para acciones prioritarias...

c. **¿Qué riesgos presentan y qué restricciones deben aportarse a su empleo?**
Existe el riesgo de que una o varias páginas sufran de inanición durante un largo periodo de tiempo porque hay uno o varios marcos bloqueados. Esto se puede solucionar aplicando un límite de tiempo el cual un marco puede estar bloqueado por una página en concreto, el cual no puede rebasar. La restricción es que solo el sistema operativo pueda utilizar este mecanismo.

23. Disponemos de un ordenador que cuenta con las siguientes características: tiene una memoria RAM de 4KBytes, permite usar memoria virtual paginada, las páginas son de 1KBytes de tamaño y las direcciones virtuales son de 16 bits. El primer marco de página (marco 0) se usa únicamente por el Kernel y los demás marcos están disponibles para su uso por los procesos que se ejecutan en el sistema. Supongamos que tenemos sólo dos procesos, P1 y P2, y que utilizan las siguientes direcciones de memoria virtual y en el siguiente orden:

Proceso	**Direcciones virtuales**
P1	0-99
P2	0-500
P1	100-500
P2	501-1500
P1	3500-6700
P2	1501-2100
P1	501-600

a. **¿Cuántos marcos de página tiene la memoria RAM de este ordenador?**
El tamaño de la memoria RAM es de 4 KB y el tamaño de las páginas es de 1KB, por tanto:

4KB/1KB = 4 marcos de página

b. **¿Cuántos bits necesitamos para identificar los marcos de página?**
2 bits

c. **Describe los fallos de página que tendrán lugar para cada intervalo de ejecución de los procesos, si la política de sustitución de páginas utilizada es LRU. Suponga que dicho algoritmo es de asignación variable y sustitución global.**
Tenemos 4 marcos disponibles, pero el marco 0 está reservado para el kernel. Por tanto, tenemos 3 marcos disponibles (1, 2, 3).

Como el tamaño de las páginas es de 1KB = 1024B, las direcciones virtuales se distribuyen de la siguiente forma:

Marco 0: [0, 1023]
Marco 1: [1024, 2047]
Marco 2: [2048, 3071]
Marco 3: [3072, 4095]

Proceso	**Direcciones virtuales**	**Marcos usados**
P1	0-99	0
P2	0-500	0
P1	100-500	0
P2	501-1500	0, 1
P1	3500-6700	3
P2	1501-2100	1,2
P1	501-600	0

La secuencia es la siguiente: *(representado como proceso-marco)*
P1-0, P2-0, P1-0, P2-0, P2-1, P1-3, P2-1, P2-2, P1-0

Siguiendo el algoritmo LRU tenemos que:

P1-0	P1-0	P1-0	P1-0	P1-0	P1-3	P1-3	P1-3	P1-0
	P2-0	P2-0	P2-0	P2-0	P2-0	P2-0	P2-2	P2-2
				P2-1	P2-1	P2-1	P2-1	P2-1
*	*			*	*		*	*

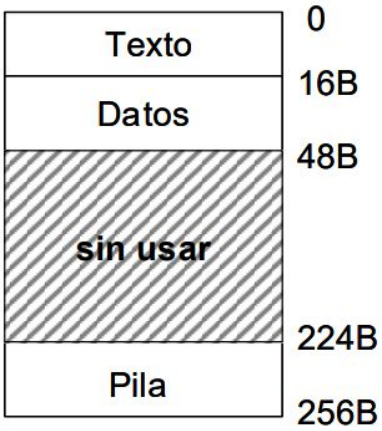
Se producen 6 faltas de página

24. Se tiene un sistema de memoria virtual con paginación a dos niveles que permite agrupar las páginas en "directorios de páginas". Cada tabla de páginas puede contener hasta 1024 páginas. Los espacios de direcciones lógicas de este sistema son de 4 Gbytes y el tamaño de página es de 4 Kbytes. El espacio de direcciones físicas puede tener hasta 1Gb. Describa la estructura de las direcciones lógicas y de las direcciones físicas de este sistema de memoria virtual.

Espacio de direcciones físicas = 1 GB = 2^{30} B → 30 bits de dirección física
Espacio de direcciones lógicas = 4 GB = 2^{32} B → 32 bits de dirección lógica
Tamaño de página = 4 KB = 2^{12} B → 12 bits de desplazamiento
Número de páginas = 1024 = 2^{10} → 10 bits para cada página

Dirección física 30 bits → 12 bits de desplazamiento + 18 bits de marco
Dirección lógica 32 bits → 12 bits de desplazamiento + 2 páginas * 10 bits (la tabla de páginas de los niveles 1 y 2 tienen 10 bits cada una)

25. Suponga un sistema que utiliza paginación a dos niveles. Las direcciones son de 8 bits con la siguiente estructura: 2 bits en la tabla de páginas de primer nivel, 2 bits en la tabla de páginas de segundo nivel y 4 bits para el desplazamiento). El espacio de direccionamiento virtual de un proceso tiene la estructura del dibujo. Represente gráficamente las tablas de páginas y sus contenidos, suponiendo que cada entrada de la tabla de páginas ocupa 8 bits y que todas las páginas están cargadas en memoria principal (elige tú mismo la ubicación en memoria principal de dichas páginas, suponiendo que la memoria principal es de 160 Bytes). Dada esa asignación traduce la dirección virtual 47.

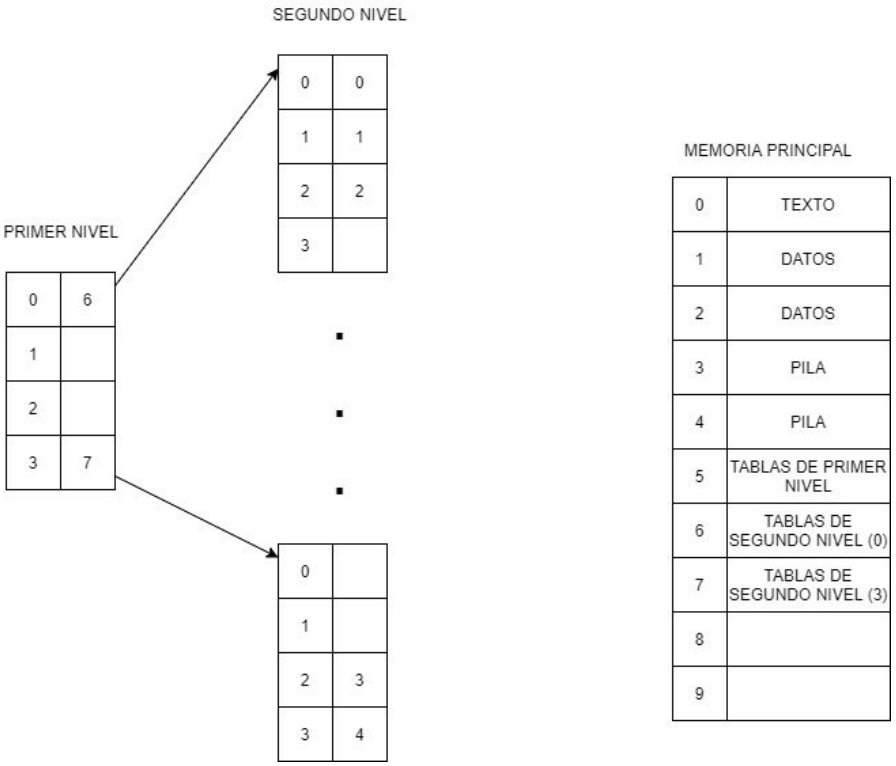


Tenemos 2 bits para codificar todas la tabla de páginas del primer nivel, por tanto, en ese nivel tenemos solo 4 páginas. El mismo razonamiento para la tabla de páginas de segundo nivel.

Como tenemos 4 bits para el desplazamiento, podemos obtener el tamaño de página:
Tamaño de página = 2^d = 2^4 = 16B

Podemos deducir, como cada página ocupa 16B, que si tenemos una memoria principal de 160B, tendremos 10 páginas en memoria principal.

Sabemos entonces que cada página puede albergar hasta 16B. Podemos plantear entonces el siguiente diagrama:



Para traducir la dirección lógica 47, tenemos que calcular varias cosas:

- Posición en la tabla de páginas del primer nivel
  (Dirección lógica entre el tamaño de la página por el número de páginas del primer nivel)
  47/(16*4) = Cociente 0, Resto 47

  Se encuentra en la primera entrada *(la entrada 0)* de la tabla de páginas del primer nivel

- Posición en la tabla de páginas del segundo nivel
  (El resto de la operación anterior entre el tamaño de página)
  47/16 = Cociente 2, resto 15

  Entrada 2 de la tabla correspondiente del segundo nivel.

- Desplazamiento
  (El resto de la operación anterior)
  47%16 = 15

26. Considere la siguiente tabla de segmentos:

Segmento	Presencia o validez	Dirección base	Longitud
0	0	219	600
1	1	2300	14
2	1	90	100
3	0	1327	580
4	1	1952	96

¿Qué direcciones físicas corresponden a las direcciones lógicas (nº_segmento, desplazamiento) siguientes? Si no puede traducir alguna dirección lógica a física, explique el por qué.

a. 0, 430
En el segmento 0 el bit de presencia está a 0, por tanto, no se puede traducir esta dirección lógica. Se produce una excepción por falta de segmento.

b. 1, 10
Dirección física = Dirección base + desplazamiento = 2300 + 10 = 2310

c. 3, 400
En el segmento 3 el bit de presencia está a 0, por tanto, no se puede traducir esta dirección lógica. Se produce una excepción por falta de segmento.

d. 4, 112
No se puede acceder porque el desplazamiento es mayor que la longitud del segmento 4. Se produce una excepción por intentar acceder a una dirección no permitida *(el clásico segmentation fault)*.

27. Respecto a la gestión de memoria que se hace en Linux: suponga que un proceso realiza una llamada al sistema fork creando un proceso hijo. Represente gráficamente como quedan las estructuras de datos relacionadas con ambos procesos.

28. ¿Qué información comparten un proceso y su hijo en un sistema Linux después de ejecutar el siguiente código? Justifique su respuesta e indique qué hace este trozo de código.

```
if  ( fork() != 0 )
    wait (&status)
else
    exec (B);
```

// usamos una llamada al sistema exec genérica con un único argumento, el nombre de un archivo ejecutable

En Linux, un proceso hijo es un clon del padre *(salvo por el PID y la memoria)*. Ambos procesos disponen de las mismas variables *(aunque estas son independientes)*. Además, el hijo hereda los descriptores de fichero del padre.

En ese trozo de código lo que ocurre es lo siguiente: Un padre crea un proceso hijo y se pone a esperar. El hijo realiza la llamada al sistema exec y ejecuta el proceso B. Cuando el proceso hijo termine desbloqueará al proceso padre.