UNIVERSIDAD DE MURCIA

Introducción a los Sistemas Operativos

Examen de teoría: problemas (2º parte del final: temas 5 y 6)
13 de enero de 2023

Apellidos:	Nombre:
Grupo: ☐ 1 ☐ 2 ☐ 3 ☐ PCEO	DNI:

Instrucciones:

- Todas las respuestas deben escribirse con **bolígrafo**.
- Se valorará la **exactitud, completitud y brevedad** de todas las respuestas, que deberán ser **RAZONADAS**. No tendrán validez las respuestas no justificadas.
- Esta parte vale 3 puntos y, por tanto, representa el 30 % de la nota del examen final de teoría.

---

1. (2 puntos) Dado un sistema con direcciones virtuales de 32 bits y físicas de 30 bits, que utiliza memoria virtual basada en paginación pura con tabla de páginas de dos niveles. Tras los 6 primeros accesos a memoria realizados por un proceso, la tabla de páginas tiene el siguiente aspecto:

Primer nivel

	MF	R	M	V
0	X'	0	0	1
1	Y'	0	0	1
…		0	0	0
		0	0	0
.		0	0	0
		0	0	0
.		0	0	0
		0	0	0
		0	0	0
511	Z'	0	0	1

0x02000000

Segundo nivel

X

	MF	R	M	V
0				0
1				0
…				0
10	0xA000	1	1	1
.				0
.				0
127	0xA001	1	0	1
				0
				0
511				0

Y

		R	M	V
0				0
1	0xA002	1	0	1
				0
				0
.				0
.				0
.				0
				0
				0
511	0xA003	1	1	1

Z

		R	M	V
0				0
1				0
…				0
16	0xA004	1	0	1
…				0
32	0xA005	1	0	1
.				0
				0
				0
511				0

El valor que aparece bajo la tabla de primer nivel es la dirección física donde se encuentra en memoria (en este caso `0x02000000`; recuerda que 30 bits necesitan 8 dígitos hexadecimales para codificarlos). En las tablas de segundo nivel puedes ver las variables X, Y y Z, que representan las respectivas direcciones de memoria donde están, mientras que las variables X', Y' y Z' son los marcos físicos correspondientes a dichas direcciones. Cada nivel de la tabla de páginas usa un único marco de página, que el SO va asignando de forma secuencial comenzando a partir de la dirección física `0x02000000` (que ya está asignada).

Responde a las siguientes cuestiones:

a) (0,1 p.) ¿Qué tamaño en bits tiene cada campo PT1 y PT2 de la dirección virtual?

b) (0,1 p.) ¿Cuál es el tamaño de página en bytes que se está usando?

c) (0,1 p.) ¿Cuántas páginas virtuales soporta la tabla de páginas?

d) (0,1 p.) ¿Qué tamaño máximo de RAM podríamos instalar?

e) (0,4 p.) El SO utiliza una política estática de asignación de marcos físicos, ha reservado inicialmente 6 marcos para el proceso empezando por el marco físico `0xA000`, y **ha ido asignando secuencialmente** marcos físicos a cada uno de los 6 accesos. Sabiendo que el campo desplazamiento de cada una de las 6 direcciones virtuales usadas ha sido 0, indica en hexadecimal y en orden cronológico **las tres primeras** direcciones virtuales a las que se ha accedido, y si ha sido un acceso de lectura o escritura.

f) (0,4 p.) Teniendo en cuenta lo indicado en el apartado anterior ¿Cuáles son los valores de las variables X, Y y Z en hexadecimal? ¿Cuáles son los valores de las variables X', Y' y Z' en hexadecimal?

g) (0,8 p.) El SO emplea el algoritmo de reemplazo NRU, seleccionando el marco físico con menor dirección en caso de empate. Indica el estado final de la tabla de páginas, con todas las direcciones en hexadecimal, tras los siguientes eventos sucedidos en orden:

- Escritura en la dirección `0x001FDAAA`
- Tic de reloj
- Lectura en la dirección `0x01004111`
- Lectura en la dirección `0xFF880CCC`
- Lectura en la dirección `0x00805998`

---

2. (0,6 puntos) Tenemos un disco duro con 16 384 cilindros, 4 cabezas, pistas de 1024 sectores y sectores de 512 bytes. Se accede a este disco utilizando bloques lógicos de 8 KiB. Con estos datos, calcula:

a) (0,3 puntos) Si la suma del tiempo promedio de latencia y el tiempo de transferencia de un *bloque lógico* es 5,15625 ms, ¿a qué velocidad gira el disco duro en RPM?

b) (0,3 puntos) Tiempo total en servir las peticiones a los siguientes bloques lógicos: 20 630, 1 413 799, 2 921, 447 293 y 849 210, si el algoritmo de planificación de disco es C-SCAN con recorrido en orden ascendente, las cabezas se encuentran inicialmente en el cilindro 845 y el tiempo de búsqueda es de $0,001$ ms por cilindro recorrido.

---

3. (0,4 puntos) Tenemos un sistema de computación con una CPU que opera a 500 MIPS y un reloj programable funcionando en modo onda cuadrada. Se sabe que el código del manejador del reloj consume el 1 % del tiempo de CPU y se compone de 100 000 instrucciones. Con estos datos, calcula:

a) (0,2 puntos) El tiempo en milisegundos entre llamadas al manejador de reloj.

b) (0,2 puntos) El valor del registro de carga del reloj si el oscilador funciona a 16 MHz.

Soluciones

Ejercicio 1:

La solución de cada apartado se muestra a continuación:

a) Los campos PT1 y PT2 tienen 9 bits cada uno. Se deduce mirando el número de elementos que tiene cada nivel de la tabla de páginas, que es 512. Para indexar 512 entradas (2^9) son necesarios 9 bits.

b) Puesto que tenemos dos niveles y cada uno de los niveles tiene 9 bits, el tamaño de página es de 14 bits, es decir, $2^{14} = 16\,384$ bytes.

c) Como los campos PT1 y PT2 tienen 9 bits cada uno, el número de página virtual tendrá $9 + 9 = 18$ bits, por tanto la tabla soporta 2^{18} páginas.

d) Puesto que tenemos 30 bits de dirección física, podremos tener un máximo de 2^{30} bytes = 1 GiBytes.

e) Las direcciones virtuales las deduciremos de los índices del primer y segundo nivel donde se encuentra cada traducción. El primer acceso llevó al marco físico `0xA000`, por lo que los índices usados son el 0 del primer nivel y el 10 del segundo, resultando en binario `0 0000 0000 | 0 0000 1010`. Si añadimos los 14 bits de desplazamiento quedaría como `0 0000 0000 | 0 0000 1010 | 00 0000 0000 0000` y poniéndolo en hexadecimal daría el valor final: `0000 0000 0000 0010 1000 0000 0000 0000`, es decir `0x00028000`. El acceso fue de escritura, puesto que está activo el bit M. De igual forma se calcularían las otras dos referencias, resultando: `0x001FC000` (Lectura) y `0x00804000` (Lectura)

f) Puesto que cada tabla de segundo nivel se encuentra en un marco físico diferente secuencialmente asignado desde la dirección `0x02000000`. Como cada marco tiene un tamaño de 16 KiB (en hexadecimal 0x4000 bytes), los valores de las variables serían: X=`0x02004000`; Y=`0x02008000` y Z=`0x0200C000`.

   Con respecto a las variables X', Y' y Z', veamos cómo se calcularían. Si ponemos X en binario (recordemos que al ser una dirección física tiene 30 bits), obtenemos la siguiente ristra de bits: `00 0010 0000 0000 0100 0000 0000 0000`. Si ahora eliminamos los 14 bits menos significativos que corresponden al desplazamiento dentro del marco físico, nos queda la siguiente ristra: `0000 1000 0000 0001`, que en hexadecimal daría X'=`0x0801`. De igual manera obtenemos Y'=`0x0802` y Z'=`0x0803`

g) Indica el estado final de la tabla de páginas tras los siguientes eventos sucedidos en orden:

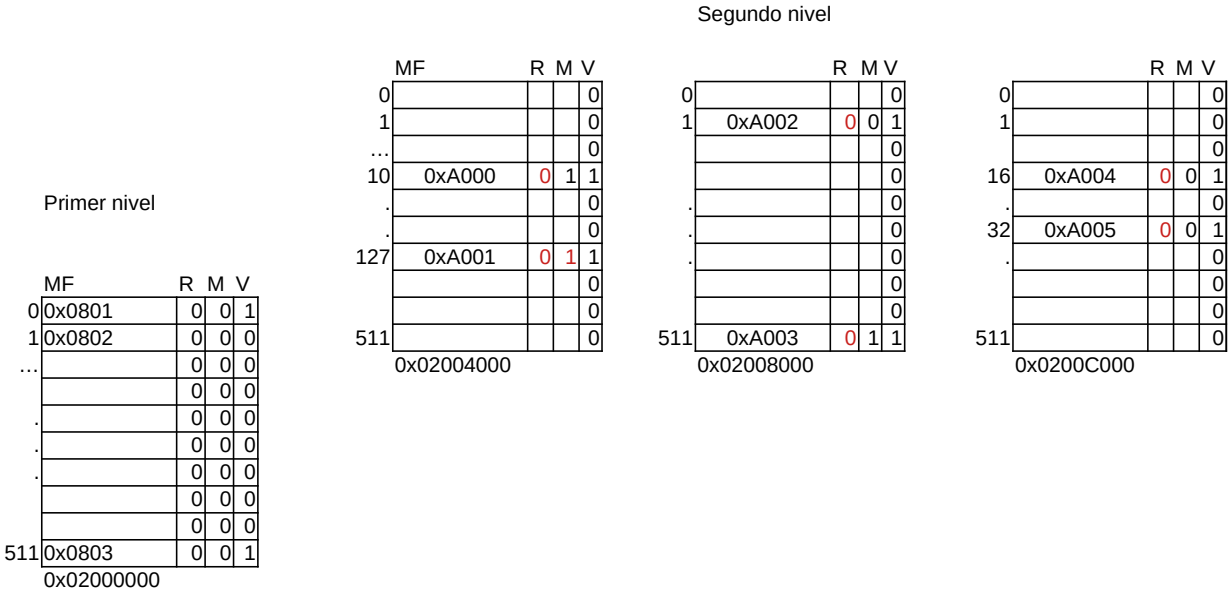
   - Escritura en la dirección `0x001FDAAA`
   - Tic de reloj
   - Lectura en la dirección `0x01004111`
   - Lectura en la dirección `0xFF880CCC`
   - Lectura en la dirección `0x00805998`

   Descompongamos cada uno de los accesos para saber los índices del primer y segundo nivel de la tabla (las direcciones virtuales son de 32 bits):

   - `0x001FDAAA` → PT1 = 0x000 (0) PT2 = 0x07F (127). En binario la dirección sería `0000 0000 0001 1111 1101 1010 1010 1010`, que si separamos en PT1, PT2 y desplazamiento y reagrupamos, nos da `0 0000 0000 | 0 0111 1111 | 01 1010 1010 1010`, por lo que PT1 = 0x000 y PT2 = 0x07F

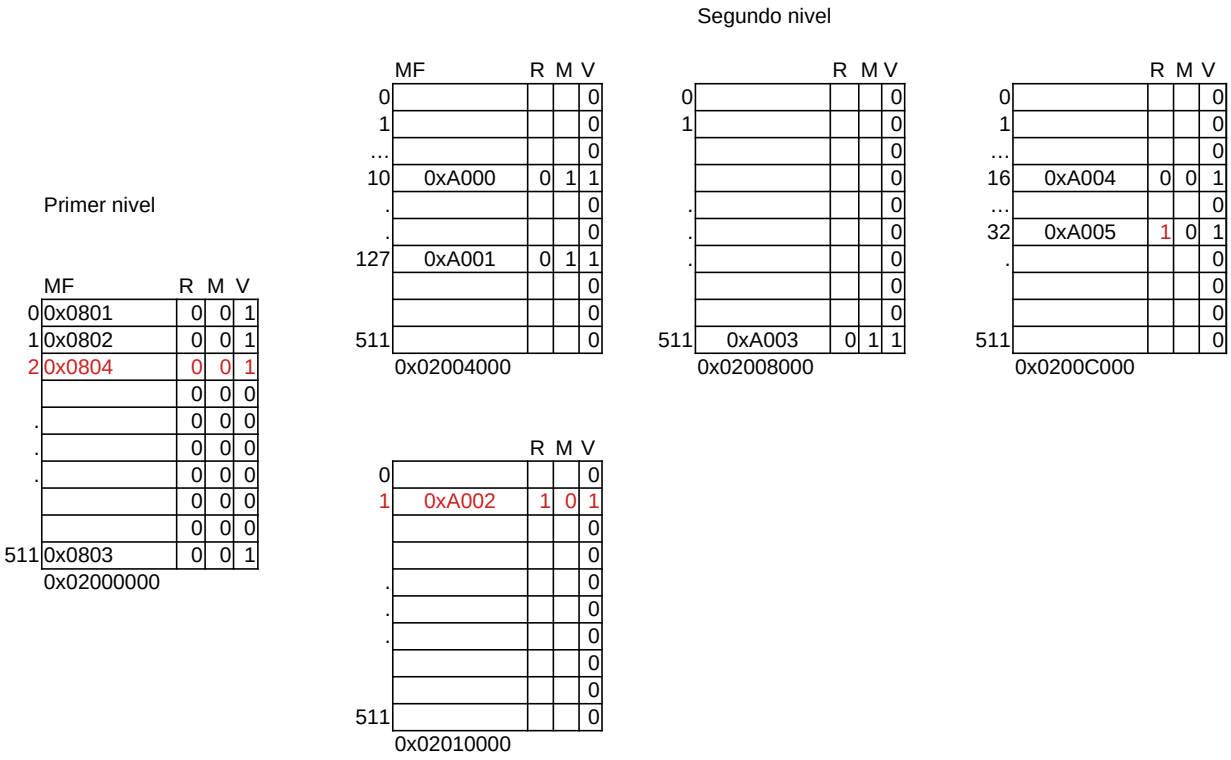
- `0x01004111` → PT1 = 0x002 (2) PT2 = 0x001 (1). En binario la dirección sería `0000 0001 0000 0000 0100 0001 0001 0001`, y haciendo lo mismo que en el caso anterior, nos da `0 0000 0010 | 0 0000 0001 | 00 0001 0001 0001`, por lo que PT1 = 0x002 y PT2 = 0x001
- `0xFF880CCC` → PT1 = 0x1FF (511) PT2 = 0x020 (32). En binario es `1111 1111 1000 1000 0000 1100 1100 1100` que dividido da `1 1111 1111 | 0 0010 0000 | 00 1100 1100 1100`
- `0x00805998` → PT1 = 0x001 (1) PT2 = 0x001 (1). En binario es `0000 0000 1000 0000 0101 1001 1001 1000` que dividido da `0 0000 0001 | 0 0000 0001 | 01 1001 1001 1000`

Ahora ya podemos ver adónde ha ido a parar cada acceso. El primer acceso ha ido a la entrada 0 del primer nivel y a la 127 del segundo, yendo a la entrada de la tabla correspondiente al marco `0xA001`, y simplemente poniendo a 1 el bit de modificado. El tic de reloj limpia todos los bit de referencia de la tabla de páginas. Tras el acceso y el tic, la tabla queda como sigue (en rojo los cambios):

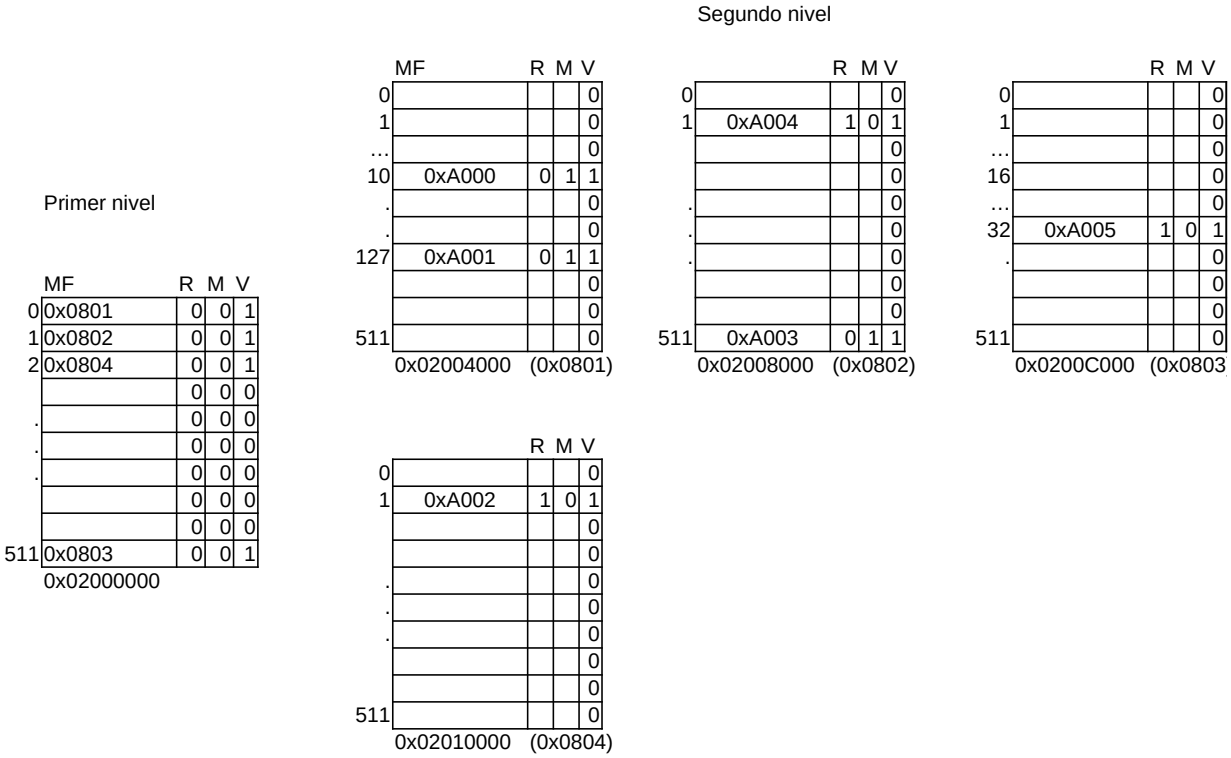


El segundo acceso a memoria va a la entrada 2 del primer nivel, que no existe, por lo que creamos una tabla de segundo nivel y el sistema operativo nos asigna el marco `0x0803`, es decir la dirección `0x02010000`. Como tenemos que asignar un marco de los 6 que tenemos a la nueva página, usamos NRU para elegir el candidato, que será el marco `0xA002`, ya que hay empate entre tres marcos que tienen el bit R=0 y M=0 (`0xA002`, `0xA004` y `0xA005`). Lógicamente hay que eliminar el marco reasignado en la entrada antigua.

El tercer acceso es un acierto y simplemente pone R=1 en la entrada correspondiente a la posición 511 del primer nivel y 32 del segundo, es decir, donde se encuentra el marco físico `0xA005`. La tabla quedaría como sigue:



En el cuarto y último acceso volvemos a acceder a la entrada 1 del primer nivel, y la entrada 1 del segundo, es decir, donde acabamos de borrar la entrada. Aplicamos NRU, y el único marco que tiene R=0 y M=0 es el `0xA004`, por lo que es el seleccionado. La tabla final queda como se muestra a continuación:



Ejercicio 2:

La solución de cada apartado se muestra a continuación:

a) Por la geometría del disco, podemos calcular que un bloque lógico, está compuesto por 16 sectores, por lo que una pista contiene $1024/16 = 64$ bloques lógicos. El tiempo medio de latencia es el tiempo que tarda el disco en dar una vuelta, y el tiempo de transferencia es el tiempo que tarda un bloque lógico en pasar por debajo de la cabeza de lectura, por tanto ambos dependen solamente del tiempo que tarda el disco en dar una vuelta (t_{vuelta}), pudiendo plantear la siguiente ecuación:

$$t_{latencia} + t_{transf} = \frac{t_{vuelta}}{2} + \frac{t_{vuelta}}{64} = 5,15625\ ms$$

Despejando podemos obtener el valor del t_{vuelta}, que sería 10 ms, por lo que $t_{latencia} = 5\ ms$ y $t_{transf} = 0,15625\ ms$. Ahora podemos ir la ecuación que calcula el tiempo de una vuelta a partir de las RPM, y obtener dichas revoluciones:

$$10\ ms = 0,01\ s = \frac{60}{RPM} \rightarrow RPM = \frac{60}{0,01} = 6000$$

b) Lo primero que necesitamos es obtener el cilindro en el que se encuentra cada bloque lógico. Como ya hemos visto, debemos hacer la división entera entre el número de bloque lógico y el total de bloques lógicos por cilindro (256):

- Bloque 20 630: cilindro 80.
- Bloque 1 413 799: cilindro 5 522.
- Bloque 2 921: cilindro 11.
- Bloque 447 293: cilindro 1 747.
- Bloque 849 210: cilindro 3 317.

Como las cabezas se encuentran inicialmente en el cilindro 845, estas peticiones (teniendo en cuenta su cilindro) se atenderán en el siguiente orden según el algoritmo C-SCAN: 1 747, 3 317, 5 522, 11 y 80. El total de cilindros recorridos será: (5 522 - 845) + (5 522 - 11) + (80 - 11) = 10 257 cilindros.

El tiempo que se tarda en recorrer todos esos cilindros es: $10\,257 \cdot 0{,}001 = 10{,}257$ ms. A este tiempo hay que sumarle el tiempo de latencia y transferencia en el que incurre cada una de las 5 peticiones. El tiempo de transferencia (t_{trans}) depende del tiempo por vuelta, del número de sectores por pista y del tamaño de bloque lógico. En enunciado ya nos habían dado la suma de los tiempos de latencia y transferencia, por lo tanto ya podemos calcular el tiempo total en servir las 5 peticiones:

$$t_{total} = t_{busqueda} + t_{latencia} + t_{transf} = 10,257\ ms + 5 \cdot 5,15625\ ms = 36,03825\ ms$$

Ejercicio 3:

La solución de cada apartado se muestra a continuación:

a) Si el código del manejador supone el 1 % del tiempo de CPU y son 100 000 instrucciones, podemos obtener el tiempo entre llamadas al manejador de la siguiente manera. Puesto que en un segundo se ejecutan 500 millones de instrucciones, el 1 % son 5 000 000 instrucciones. Y puesto que el manejador ejecuta 100 000 instrucciones, podemos deducir fácilmente que dicho manejador se ejecuta $5\,000\,000/100\,000 = 50$ veces por segundo, dando un periodo de 20 milisegundos.

b) Puesto que tenemos un oscilador a 16 MHz y se producen 50 tics por segundo, simplemente tenemos que dividir para obtener el valor del registro: $16\,000\,000/50 = 320\,000$