

SISTEMAS OPERATIVOS. Grado en Informática. TGR: Memoria y Memoria Virtual

1. En un sistema la memoria tiene un tiempo de acceso de 50 ns, el TLB de 5 ns, y el disco utilizado para paginación tiene un tiempo medio de búsqueda de 3 ms, una latencia de 1ms y una velocidad de transferencia de 4 Mbytes/seg. Sabiendo que el tamaño de página es de 4K, calcular el tiempo de acceso efectivo en cada uno de los siguientes casos:
 - a) Sin paginación
 - b) Con paginación y sin TLB
 - c) Con paginación y una probabilidad de acierto en el TLB del 80%
 - d) Con paginación, una probabilidad de acierto en el TLB del 100% y una probabilidad de fallo de página de 10^{-6}
2. A continuación se muestra la salida del comando pmap para un proceso

```
#) pmap 5396
5396: ./a.out
0000000000400000      56K r-x-- a.out
000000000060e000       4K rw--- a.out
000000000060f000       16K rw--- [ anon ]
000000000130d000      132K rw--- [ anon ]
00007f933f986000      1140K ----- file.tar
00007f933fad3000     500000K rw-s-  [ shmid=0x45e0022 ]
00007f935e31b000     500000K rw-s-  [ shmid=0x45e0022 ]
00007f937cb63000    2441408K rw--- [ anon ]
00007f9411b93000    1660K r-x-- libc-2.19.so
00007f9411d32000    2048K ----- libc-2.19.so
00007f9411f32000    16K r---- libc-2.19.so
00007f9411f36000     8K rw--- libc-2.19.so
00007f9411f38000    16K rw--- [ anon ]
00007f9411f3c000    128K r-x-- ld-2.19.so
00007f9412000000    1140K ----- file.tar
00007f941211d000    12K rw--- [ anon ]
00007f9412140000    24K rw-s-  [ shmid=0x4608024 ]
00007f9412146000    24K rw-s-  [ shmid=0x45f0023 ]
00007f941214c000    24K rw-s-  [ shmid=0x4608024 ]
00007f9412152000    24K rw-s-  [ shmid=0x45f0023 ]
00007f9412158000    16K rw--- [ anon ]
00007f941215c000     4K r---- ld-2.19.so
00007f941215d000     4K rw--- ld-2.19.so
00007f941215e000     4K rw--- [ anon ]
00007ffcb7251000   132K rw--- [ stack ]
00007ffcb72dd000     8K r-x-- [ anon ]
00007ffcb72df000     8K r---- [ anon ]
ffffffffff600000     4K r-x-- [ anon ]
total                 3448060K
```

- a) ¿Se trata de un sistema de 32 bits o 64 bits?
- b) ¿Se ha compilado estáticamente?
- c) ¿Cuanto ocupa el proceso en memoria?
- d) ¿Qué hay en la dirección de memoria física 0x7f9412000000?
- e) ¿Utiliza alguna región de memoria compartida? ¿cuántas?

3. Un sistema de 16 bits tiene páginas de 1 Kbytes y 32K de memoria instalada. Cada entrada de la tabla de páginas consta de 16 bits, cuyo significado es el siguiente

- **bits 0-5:** Número de marco físico, (6 bits más significativos de la dirección de memoria física)
- **bit 6:** Protección: 1, (r/w), 0 (ro)
- **bit 7:** Privilegio (1, página solo accesible en modo kernel)
- **bit 8:** Presencia (1, página en memoria)
- **bits 9:** Referencia (1, página referenciada)
- **bits 10-15:** 000000, entrada no válida; 111111, entrada válida

En dicho sistema en un instante dado hay, aparte del S.O., un proceso, P_1 cuya tabla de páginas se muestra a continuación

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | ... |
|-------|------|------|------|------|------|------|------|------|------|
| P_1 | FF1C | FF1D | FF1E | FC40 | FC4F | FC4E | FF46 | 0000 | ... |
| | ... | 38 | 39 | 3A | 3B | 3C | 3D | 3E | 3F |
| P_1 | ... | 0000 | FC50 | FF4C | FF50 | FF80 | FF81 | FFC2 | FFC3 |

- a) ¿Tiene dicho sistema memoria virtual?
- b) Muéstrese cómo está dicho proceso en memoria
- c) ¿Cuánta memoria libre queda en el sistema?
- d) ¿Cuál es el tamaño virtual del proceso P_1 ? ¿Cuánto ocupa en memoria?
- e) ¿En qué dirección virtual comienzan los datos y la pila del proceso?
- f) ¿En qué dirección virtual ve el proceso el S.O.? ¿Dónde está el S.O. en la memoria física?
- g) ¿Qué ocurriría si P_1 intenta leer los contenidos de la posición de memoria 0x04F9? ¿A qué posición de memoria acceden realmente? ¿Y si intentan escribir en dicha posición?
- h) ¿Qué ocurriría si P_1 intentan leer los contenidos de la posición de memoria 0xE42A? ¿A qué posición de memoria se accede realmente?
- i) ¿Podría implementarse LRU en este sistema?
- j) ¿Podría P_1 tener declarada una variable local de `main char a[3300]`? ¿Y si fuese externa?

4. ¿Por qué se utilizan tablas de páginas en varios niveles?

5. ¿Qué es la anomalía de Belady y que tipo de algoritmos de reemplazo no la presentan
6. Considérese las siguiente cadena de referencias a memoria (donde cada letra representa una página) ABCDAABCCDDDDDEFGEFGEFGAAAD. Indíquese cuantos fallos de página hay y muéstrese el conjunto residente con cada referencia utilizando
 - a) Working Set de ventana 4. Número de fallos de página: ____
 - b) L.R.U. con 4 marcos. Número de fallos de página: ____
7. Un sistema de 16 bits tiene paginación con memoria virtual, El tamaño de página es de 1K y cada entrada de la tabla de páginas tiene 8 bits, cuyo significado es el siguiente
 - **bits 0-5:** Número de marco físico, (6 bits mas significativos de la dirección de memoria física)
 - **bit 6:** Protección: 1, (r/w), 0 (ro)
 - **bit 7:** Privilegio (1, pagina solo accesible en modo kernel)
 - **bit 8:** Presencia (1, página en memoria)
 - **bits 9:** Referencia (1, página referenciada)

- **bits 10-15:** 000000, entrada no válida; 111111, entrada válida

En un instante dado la memoria física del sistema está como se muestra a continuación:

| | |
|----------|--------------|
| $0x4000$ | P_1 STACK2 |
| $0x3C00$ | |
| $0x3800$ | P_1 CODE2 |
| $0x3400$ | |
| $0x3000$ | P_1 STACK1 |
| $0x2C00$ | P_1 DATA2 |
| $0x2800$ | |
| $0x2400$ | |
| $0x2000$ | P_1 CODE0 |
| $0x1C00$ | P_1 DATA0 |
| $0x1800$ | P_1 DATA1 |
| $0x1400$ | P_1 DATA3 |
| $0x1000$ | |
| $0x0C00$ | S.O. (data) |
| $0x0800$ | S.O. (data) |
| $0x0400$ | S.O. (code) |
| $0x0000$ | S.O. (code) |

- ¿Cuánta memoria hay instalada en el sistema?
- ¿Cuánta hay libre?
- Sabiendo que en dicho sistema los procesos ven el S.O. en las direcciones virtuales más altas, y que P_1 tiene 3 páginas de código (CODE0, CODE1 y CODE2) que comienzan en la dirección virtual $0x0000$, 4 páginas de datos (DATA0, DATA1, DATA2 y DATA3) que comienzan en la dirección virtual $0x0C00$, y 3 páginas de pila (STACK0, STACK1 y STACK2) que comienzan en la dirección virtual $0xE400$. Construir la tabla de páginas para dicho proceso en dicho instante.

Ejercicio 1.

$$TA = 50\text{ns}$$

$$TLB = 5\text{ns}$$

disco paginación \Rightarrow $TMB = 3\text{ms}$, $\text{Lat} = 1\text{ms}$, $Vtrans = 4\text{Mbytes/s}$

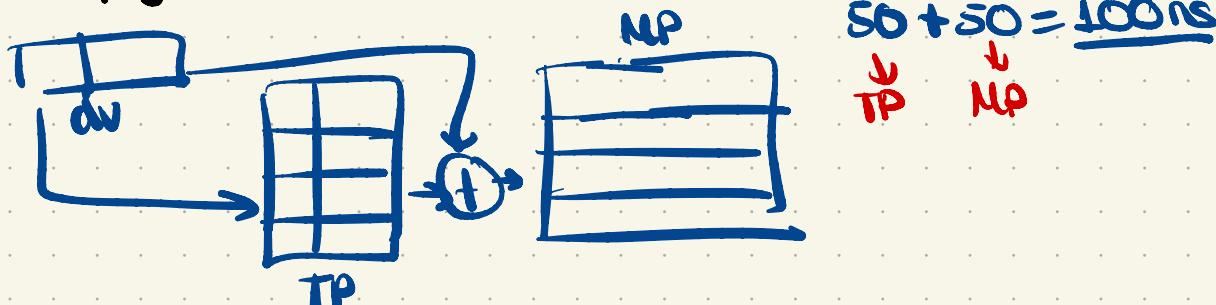
$$T/\text{página} = 4\text{KB}$$

T acceso efectivo en:

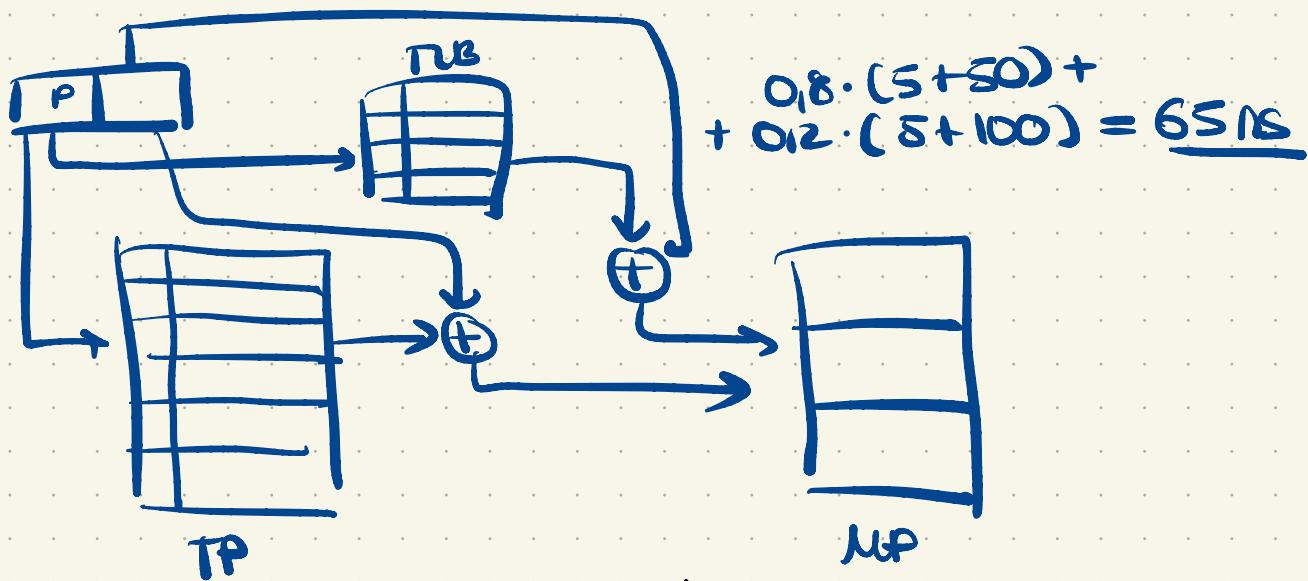
a) Sin paginación.

$50\text{ns} \Rightarrow$ sin paginación o segmentación no hay TLB

b) Con paginación sin TLB.



c) Con paginación y probabilidad de acierto de TLB de 80%.



d) Con paginación, una probabilidad de acierto de TLB de 100% y probabilidad de fallo de página de 10^{-6} .

Para transferir 1 pág de 4KB emplea

$$4 \cdot 2^{20} \text{B} \rightarrow 1\text{s} \text{ y } 0,001 \text{ s} \approx 1\text{ms}$$

$$4 \cdot 2^{10} \text{B} \rightarrow x$$

$$t \text{ servicio de fallo de página} = 5\text{ms}$$

$$0,999999 \cdot 55 + \underbrace{10^{-6} \cdot 5\text{ms}}_{5\text{ns}} \approx 60\text{ ns}$$

Ejercicio 2

- a) Sistema de 64 bits \Rightarrow 4 bits / dígito hexadecimal en las direcciones.
- b) No. se ha compilado de forma dinámica porque hay lib.so y ld.so y no con extensión .a
- c) No puede saberse, pmap nos muestra el espacio virtual.
- d) No puede saberse, pmap nos muestra el espacio virtual.
- e) Sí. Se utilizan 3 regiones, cada una de ellas mapeada 2 veces.

Ejercicio 3

16 bits/dirección

1KB/página \rightarrow 16 bits/dirección

32KB memoria

- bits 0-5: Número de marco físico, (6 bits más significativos de la dirección de memoria física)
- bit 6: Protección: 1, (r/w), 0 (ro)
- bit 7: Privilegio (1, página solo accesible en modo kernel)
- bit 8: Presencia (1, página en memoria)
- bits 9: Referencia (1, página referenciada)
- bits 10-15: 000000, entrada no válida; 111111, entrada válida

En dicho sistema en un instante dado hay, aparte del S.O., un proceso, P_1 cuya tabla de páginas se muestra a continuación

| | | | | | | | | | |
|-------|------|------|------|------|------|------|------|------|------|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | ... |
| P_1 | FF1C | FF1D | FF1E | FC40 | FC4F | FC4E | FF46 | 0000 | ... |
| | ... | 38 | 39 | 3A | 3B | 3C | 3D | 3E | 3F |
| P_1 | ... | 0000 | FC50 | FF4C | FF50 | FF80 | FF81 | FFC2 | FFC3 |

- a) Sí tiene memoria virtual porque utiliza bit de presencia y a la vista de las entradas de la tabla de páginas sí se utiliza.

b) Asumimos que dirección base es 0

entrada 0: FF1C $\begin{array}{ccccccccc} \text{en mem} & \downarrow & \text{0x1C} \\ \text{0x} & \text{7} & \text{0} & \text{0001} & \text{1100} \\ \text{0111} & \text{0000} & \text{0000} & \text{0000} & \text{0000} \end{array}$

entrada 1: FF1D $\begin{array}{ccccccccc} \text{en mem} & \downarrow & \text{0x1D} \\ \text{0x} & \text{7} & \text{4} & \text{0001} & \text{1101} \\ \text{0111} & \text{0100} & \text{0000} & \text{0000} & \text{0000} \end{array}$

entrada 2: FF1E $\begin{array}{ccccccccc} \text{en mem} & \downarrow & \text{0x1E} \\ \text{0x} & \text{7} & \text{8} & \text{0001} & \text{1110} \\ \text{0111} & \text{1000} & \text{0000} & \text{0000} & \text{0000} \end{array}$

entrada 3: FC40 $\begin{array}{ccccccccc} \text{en mem} & \downarrow & \text{0x40} \\ \text{0x} & \text{8} & \text{0} & \text{0100} & \text{0000} \end{array}$

entradas 4,5 : FC4F, FC4E no en mem

entrada 6 : FF46 $\begin{array}{ccccccccc} \text{en mem} & \downarrow & \text{0x46} \\ \text{0x} & \text{8} & \text{0} & \text{0100} & \text{0110} \\ \text{0001} & \text{1000} & \text{0000} & \text{0000} & \text{0000} \end{array}$

entradas 7-38 : no válidas

entrada 3A : no en mem

entrada 3A: FF4C $\begin{array}{ccccccccc} \text{en mem} & \downarrow & \text{0x4C} \\ \text{0x} & \text{3} & \text{0} & \text{0100} & \text{1100} \\ \text{0011} & \text{0000} & \text{0000} & \text{0000} & \text{0000} \end{array}$

entrada 3B : FF50 $\begin{array}{ccccccccc} \text{en mem} & \downarrow & \text{0x50} \\ \text{0x} & \text{4} & \text{0} & \text{0101} & \text{0000} \\ \text{0100} & \text{0000} & \text{0000} & \text{0000} & \text{0000} \end{array}$

entrada 3C : FF80 $\begin{array}{ccccccccc} \text{en mem} & \downarrow & \text{0x80} \\ \text{0x} & \text{0} & \text{0} & \text{1000} & \text{0000} \\ \text{0000} & \text{0000} & \text{0000} & \text{0000} & \text{0000} \end{array}$

entrada 3D: FF81 1111 1111 1000 0001

0x 0000 0100 0000 0000
0x 0 4 0 0

entrada 3G: FFC2 1111 1111 1100 0000

0x 0000 1000 0000 0000
0x 0 8 0 0

entrada 3F: FFC3 1111 1111 1100 0011

0x 0000 1100 0000 0000
0x 0 C 0 0

| | |
|---------|-------|
| 0x7100 | / |
| 0x7800 | 2 |
| 0x74000 | 1 |
| 0x7000 | 0 |
| ... | Libre |
| 0x4000 | 3B |
| ... | 3entr |
| 0x3000 | 3A |
| ... | 5entr |
| 0x1800 | 6 |
| 0x1400 | / |
| 0x1000 | / |
| 0x0C00 | 3F |
| 0x0800 | 3G |
| 0x0400 | 3D |
| 0x0000 | 3C |

$$1KB = 2^{10} B$$

0000 0100 0000 0000 = 0x400
Las direcciones deben ser múltiplos de 0x400

c) Memoria libre

$$22 \text{ bloques libres} = 22 \cdot 1KB = \\ = 22 KB \text{ libres.}$$

d) Tamaño virtual del proceso y espacio ocupado en memoria.

10 páginas. 64bytes.

e) Datos: pág 3 = dv 0000 1100 0000 0000
0x0C00

Pila: pág 39 = dv 1110 0100 0000 0000
0xE400

f) dv. 80.

g)

Ejercicio 6

L.R.U. con 4 marcos

b)

| | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | A | A | B | B | C | C | D | D | D | D | D | E | F | G | E | F | G | E | F | G | A | A | A | D |
| A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A |
| B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B | B |
| C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C | C |
| D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D |

| | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | D | A | B | B | C | C | D | D | D | D | D | D | E | F | F | F | F | F | F | A | A | A | A | |
| x | A | B | C | D | A | A | B | B | C | C | D | D | D | D | D | x | D | F | F | E | E | E | E | E | E | E | D |
| x | A | B | B | C | D | A | A | B | B | C | C | D | D | D | D | x | D | G | G | E | E | E | E | E | E | E | F |
| x | A | A | B | C | D | A | A | B | B | C | C | D | D | D | D | x | D | G | G | E | E | E | E | E | E | E | F |
| x | A | A | A | B | C | A | A | B | B | C | C | D | D | D | D | x | D | G | G | E | E | E | E | E | E | E | F |

Ejercicio 7

c)

0x4000, 10100 0000 0000 0000

1111 1111 0101 0000

0x F F S O

0x E000, 1110 0100 0000 0000

0x 0C00, 0000

0x 0000, 10000 0000 0000 0000

1111 1111 1000 0000

F F 4 O

OC → 12

001100

SISTEMAS OPERATIVOS. Grado en Informática.

TGR: Memoria y Memoria Virtual

SOLUCIÓN:

1. a) **50 ns**

- b) **100 ns**

50 para acceder a la tabla de páginas que está en memoria principal + 50 para el acceso a memoria

- c) **65 ns**

Si hay acierto en el TLB son 55 ns (5 para el TLB + 50 el acceso a memoria). Si no hay acierto en el TLB son 105 ns (5 acceso al TLB + 50 acceso a la tabla de páginas en memoria + 50 acceso a memoria).

El tiempo efectivo, con una probabilidad del 80% de acierto en el TLB es $0'8 \times 55 + 0'2 \times 105 = 65$

- d) **60 ns**

El tiempo de servicio del fallo de página lo podemos aproximar al tiempo de lectura de la página desde disco, es decir búsqueda + latencia + transferencia. Puesto que el disco transfiere 4Mbytes/seg, para transferir una página (4K) emplea aproximadamente 1ms. Por tanto el tiempo de servicio del fallo de página es de 5 ms (búsqueda + latencia + transferencia)

El tiempo efectivo de acceso a memoria es $0'999999x55ns + 10^{-6}x5ms \simeq 60ns$

2. a) 64 bits, todas las direcciones se representan con 16 cifras hexadecimales
- b) No, se ven los segmentos de *libc* y *ld*
- c) No puede saberse, *pmap* nos muestra el espacio virtual
- d) No puede saberse, *pmap* nos muestra el espacio virtual
- e) Si, utiliza tres regiones distintas, mapeada cada una de ellas 2 veces

3. a) La tabla de páginas de dicho sistema sí proporciona bit de presencia, y a la vista de las entradas de la tabla de páginas vemos que se utiliza, por lo tanto: SÍ
- b) Comenzaremos analizando las entradas de la tabla de páginas de P_1

página 0 El valor de la entrada es FF1C (1111 1111 0001 1100). Los primeros 6 bits representan que la entrada es válida, el bit de presencia es 1, con lo que la página está en memoria. La página es de solo lectura (bit 6 es 0), ha sido referenciada (bit 9 es 1). Como es de solo lectura y accesible en modo usuario (bit 7 a 0), es código. Por tanto, la primera página es código y está en memoria en el marco 0x1C (dirección 0111 0000 0000 0000, (0x7000) P_1 CODE0)

página 1 El valor de la entrada es FF1D (1111 1111 0001 1101). Es una entrada válida, presente en memoria, accesible en modo usuario en solo lectura. Es, por tanto, la segunda de las páginas de código del P_1 , que denominaremos P_1 CODE1. Los 6 bits más significativos de la dirección del marco físico son 01 1101, por tanto la página está en la dirección física 0111 0100 0000 0000, (0x7400)(P_1 CODE1)

página 2 FF1E (1111 1111 0001 1110) válida, presente en memoria y referenciada solo lectura y accesible en modo usuario: código en 0111 1000 0000 0000 (0x7800) (P_1 CODE2)

página 3 FC40 (1111 1100 0100 0000) válida. lectura/escritura en modo usuario (bit 7 a 0 y bit 6 a 1), por tanto datos. pero NO PRESENTE EN MEMORIA. Puesto que no está en memoria la dirección del marco es irrelevante.

página 4 FC4F. (1111 1100 0100 1111) válida, lectura/escritura en modo usuario (bit 7 a 0 y bit 6 a 1) por tanto datos pero NO PRESENTE EN MEMORIA

- página 5 FC4E. (1111 1100 0100 1110) igual que la anterior.
- página 6 FF46, (1111 1111 0100 0110) Datos de usuario en el marco 001100 es decir en la dirección 0001 1000 0000 0000 (0x1800) (P_1 DATA3).
- págs 7..38 Direcciones no válidas: “huecos” en el espacio virtual de direcciones del proceso
- página 39 FC50 (1111 1100 0101 0000) válida, No presente en memoria lectura/escritura en modo usuario (pila pues en la parte alta del espacio de direcciones)
- página 3A FF4C (1111 1111 0100 1100) válida. Presente y referenciada. Lectura/escritura en modo usuario en la dirección física 0011 0000 0000 0000 (0x3000). Esta en la parte alta del espacio virtual de direcciones: pila de usuario (P_1 STACK1)
- página 3B FF50 (1111 1111 0101 0000) válida. Presente y referenciada. Lectura/escritura en modo usuario en la dirección física 0100 0000 0000 0000 (0x4000) (P_1 STACK2)
- página 3C FF80 (1111 1111 1000 0000) válida. Presente y referenciada. Solo lectura en modo kernel: código del S.O. en 0x0000
- página 3D FF81 (1111 1111 1000 0001) válida. Presente en memoria, referenciada, código del S.O. en 0000 0100 0000 0000 (0x0400)
- página 3E FFC2 (1111 1111 1100 0010) válida. Presente en memoria, referenciada, lectura/escritura, accesible sólo en modo kernel: Datos del S.O. en dirección 0x0800
- página 3F FFC3 (1111 1111 1100 0011) válida. Presente en memoria, referenciada, lectura/escritura, accesible sólo en modo kernel: Datos del S.O. en dirección 0x0C00

Como las páginas son de 1K, cada marco

comienza en una dirección múltiplo de 0x400.

| | |
|--------|---------------|
| 0x7C00 | |
| 0x7800 | P_1 CODE2 |
| 0x7400 | P_1 CODE1 |
| 0x7000 | P_1 CODE0 |
| 0x6C00 | |
| 0x6800 | |
| 0x6400 | |
| 0x6000 | |
| 0x5C00 | |
| 0x5800 | |
| 0x5400 | |
| 0x5000 | |
| 0x4C00 | |
| 0x4800 | |
| 0x4400 | |
| 0x4000 | P_1 STACK 2 |
| 0x3C00 | |
| 0x3800 | |
| 0x3400 | |
| 0x3000 | P_1 STACK 1 |
| 0x2C00 | |
| 0x2800 | |
| 0x2400 | |
| 0x2000 | |
| 0x1C00 | |
| 0x1800 | P_1 DATA 3 |
| 0x1400 | |
| 0x1000 | |
| 0x0C00 | S.O. (data) |
| 0x0800 | S.O. (data) |
| 0x0400 | S.O. (code) |
| 0x0000 | S.O. (code) |

- c) A la vista de la representación, quedan libres 22 marcos de 1Kbytes, es decir, 22 Kbytes
- d) El proceso P_1 puede referenciar (excluyendo el S.O.) 10 páginas (3 de código, 4 de datos y 3 de pila). En memoria P_1 ocupa 6 Kbytes (6 páginas con el bit de presencia)
- e) Para P_1 sus datos comienzan en la página 3, es decir. dirección virtual 0000 1100 0000

0000 0x0C00, mientras la pila comienza en la página 0x39, 1110 0100 0000 0000 es decir, dirección virtual 0xE400. es decir 0xD800.

- f) Página 3E los datos del kernel (dirección virtual 1111 1000 0000 0000, 0xF800) y página 0x3C el código del kernel (dirección 1111 0000 0000 0000 0xF000)
 - g) 0x04F9 0000 0100 1111 1001, es decir página 1 (000001) desplazamiento 0F9 (00 1111 1011). Como dicha página está en el marco 0x7400 se accede a la dirección 0x74F9. Un intento escritura produciría una excepción, pues la página es de solo lectura
 - h) 0xE42A es 1110 0100 0010 1010, es la página 0x39 (111001) desplazamiento 2A (00 0010 1010). La página no está en memoria y se produciría una excepción de fallo de página. No sabemos a que dirección de memoria física se accede pues depende de donde colo el S.O. la página al servir el fallo de página
 - i) LRU no puede implementarse con solo un bit de referencia
 - j) Dicha variable ocupa 3300 bytes, es decir, más de tres páginas. La variables locales se almacenan en la pila, la pila de P_1 tiene 3 páginas (3072 bytes), por tanto no. Si fuese externa, dado que P_1 tiene cuatro páginas de datos, sí podría ser.
4. En espacios virtuales muy grandes, no es necesario asignar espacio para tablas de páginas en los huecos del espacio de direcciones, utilizando menos memoria para las tablas de páginas que en el caso de usar T.P. en un nivel.
 5. Se llama anomalía de Belady a la situación que puede presentarse en algunos algoritmos de reemplazo de página, con algunas cadenas de referencia concretas, en la que al aumentar el número de marcos asignado aumenta el número de fallos de página. FIFO presenta dicha anomalía, LRU y óptimo garantizan que no. Los algoritmos que no la presentan son los denominados

algoritmos de pila, algortimos en los cuales el conjunto residente con N marcos es un subconjunto del conjunto residente con N+1 marcos

6.-

Supondremos que inicialmente no hay ninguna página en memoria.

Indicamos con * donde hay fallo de página

L.R.U. con 4 marcos

| | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | A | A | B | B | C | C | D | D | D | D | E | F | G | E | F | G | E | F | G | A | A | A | D |
| * | * | * | * | * | | | | | | | | | | * | * | * | | | | | | | | * | | * |
| A | A | A | A | A | A | A | A | A | A | A | A | A | A | E | E | E | E | E | E | E | E | E | E | E | E | D |
| B | B | B | B | B | B | B | B | B | B | B | B | B | B | F | F | F | F | F | F | F | F | F | F | F | F | F |
| C | C | C | C | C | C | C | C | C | C | C | C | C | C | G | G | G | G | G | G | G | G | G | G | G | G | |
| D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | D | A | A | A | |

Working Set de Ventana 4. Nótese que la fila no indica el marco, sino que sólo debe considerarse el conjunto residente en memoria. El símbolo '=' indica que se mantiene el conjunto de la referencia anterior.

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | A | A | B | B | C | C | D | D | D | D | E | F | G | E | F | G | E | F | G | A | A | A | D | | |
| * | * | * | * | * | | * | | * | | * | | | | * | * | * | | | | | | | | * | | * | | |
| A | B | C | D | = | A | B | B | C | C | D | D | = | D | D | E | F | G | E | = | = | = | = | = | = | A | A | A | D |
| A | B | C | = | D | A | A | B | B | C | C | = | D | D | D | E | F | G | E | = | = | = | = | = | = | A | G | G | A |
| A | B | C | D | A | C | D | A | B | A | B | C | B | | | | | | | | | | | | | F | E | | |

7.-

a) Hay 17 Kbytes instalados (17 páginas de 1K)

b) 5Kbytes (el S.O. ocupa 4K y P₁ 8Kbytes)

c) El proceso tiene tres páginas de código que comienzan en la dirección virtual 0x0000 es decir son la página 0, la página 1 y la página 2.

Tiene 4 páginas de datos que comienzan en la dirección virtual 0x0c00 esto es 0000 1100 0000 0000 y por tanto corresponde a las páginas 3, 4, 5 y 6.

Tiene tres páginas de pila que comienza ne la dirección virtual 0xE400 1110 0100 0000 0000, por tanto las páginas de pila son las páginas 39, 3Ay 3B. Si el sistema operativo está en las direcciones más altas del espacio virtual (del enunciado se ve que son 4 páginas) esto son losas páginas 3C, 3D, 3E y 3F

Como el enunciado no da más datos, suponemos que las páginas que están en memoria han sido referenciadas. Construimos ahora la tabla de páginas a partir del diagrama de la memoria del sistema.

| Entrada Página | Bits 10-15 (Valida) | Bit 9 (R) | Bit 8 (P) | Bit 7 K/U | Bit 6 R/W | Bits 0-5 (n marco) | Valor entrada Binario (HEX) |
|----------------|------------------------|--------------|--------------|--------------|--------------|-----------------------|-------------------------------------|
| 3F (SO 3) | 111111 | 1 | 1 | 1 | 1 | 3(000011) | 1111 1111 1100 0011 (0xFFC3) |
| 3E (SO 2) | 111111 | 1 | 1 | 1 | 1 | 2(000010) | 1111 1111 1100 0010 (0xFFC2) |
| 3D (SO 1) | 111111 | 1 | 1 | 1 | 0 | 1(000001) | 1111 1111 1000 0001 (0xFF81) |
| 3C (SO 0) | 111111 | 1 | 1 | 1 | 0 | 0(000000) | 1111 1111 1000 0000 (0xFF80) |
| 3B (STK2) | 111111 | 1 | 1 | 0 | 1 | 16(010000) | 1111 1111 0101 0000 (0xFF50) |
| 3A (STK1) | 111111 | 1 | 1 | 0 | 1 | 12(001100) | 1111 1111 0100 1100 (0xFF4C) |
| 39 (STK0) | 111111 | 0 | 0 | 0 | 1 | (???????) | 1111 1100 01?? ???? |
| ... | 000000 | 0 | 0 | 0 | 0 | 00000 | 0000 0000 0000 0000 (0x0000) |
| ... | 000000 | 0 | 0 | 0 | 0 | 00000 | 0000 0000 0000 0000 (0x0000) |
| | 000000 | 0 | 0 | 0 | 0 | 00000 | 0000 0000 0000 0000 (0x0000) |
| 6 (DATA3) | 111111 | 1 | 1 | 0 | 1 | 5(000101) | 1111 1111 0100 0101 (0xFF45) |
| 5 (DATA2) | 111111 | 1 | 1 | 0 | 1 | 11(001011) | 1111 1111 0100 1011 (0xFF4B) |
| 4 (DATA1) | 111111 | 1 | 1 | 0 | 1 | 6(000110) | 1111 1111 0100 0110 (0xFF46) |
| 3 (DATA0) | 111111 | 1 | 1 | 0 | 1 | 7(000111) | 1111 1111 0100 0111 (0xFF47) |
| 2 (CODE2) | 111111 | 1 | 1 | 0 | 0 | 14(001110) | 1111 1111 0000 1110 (0xFF0E) |
| 1 (CODE1) | 111111 | 0 | 0 | 0 | 0 | ?????? | 1111 1100 00?? ???? |
| 0 (CODE0) | 111111 | 1 | 1 | 0 | 0 | 8(001000) | 1111 1111 0000 1000 (0xFF08) |