

# Tema 7. Memòria Virtual Problemes

Curs 2019-20 Primavera

Grup 30

Joan Manuel Parcerisa



UNIVERSITAT POLITÈCNICA DE CATALUNYA  
BARCELONATECH  
Facultat d'Informàtica de Barcelona



# Problema 7.1

**7.1.** Es disposa d'un sistema amb memòria virtual paginada, amb les característiques següents:

- Espai d'adreces dels programes (grandària de la memòria virtual)= 16 Mbytes =  $2^{24}$  bytes
- Espai de memòria física= 16 Kbytes =  $2^{14}$  bytes
- Mida de les pàgines = 256 bytes =  $2^8$  bytes

Cada programa pot tenir a memòria física un màxim de 3 pàgines simultàniament. En cas que una d'aquestes 3 pàgines hagués de ser sacrificada per deixar lloc a una altra, es fa servir l'algorisme de reemplaçament LRU.

Es demana:

- a) Indica les dimensions de la taula de pàgines de cada programa, tenint en compte que cada entrada de la taula té un bit de presència (indica si la pàgina corresponent està carregada a memòria física) i un bit de modificació (indica si la pàgina corresponent ha estat modificada durant la seva estada a la memòria física).

# Problema 7.1

7.1. Es disposa d'un sistema amb memòria virtual paginada, amb les característiques següents:

- Espai d'adreces dels programes (grandària de la memòria virtual) = 16 Mbytes =  $2^{24}$  bytes
- Espai de memòria física = 16 Kbytes =  $2^{14}$  bytes
- Mida de les pàgines = 256 bytes =  $2^8$  bytes

Cada programa pot tenir a memòria física un màxim de 3 pàgines simultàniament. En cas que una d'aquestes 3 pàgines hagués de ser sacrificada per deixar lloc a una altra, es fa servir l'algorisme de reemplaçament LRU.

Es demana:

- Indica les dimensions de la taula de pàgines de cada programa, tenint en compte que cada entrada de la taula té un bit de **presència** (indica si la pàgina corresponent està carregada a memòria física) i un bit de **modificació** (indica si la pàgina corresponent ha estat modificada durant la seva estada a la memòria física).
- Quants marcs de pàgina té l'espai físic?
    - $2^{14}$  bytes totals /  $2^8$  bytes per pàgina =  $2^6$  marcs → PPN té **6 bits**
  - Quantes pàgines té l'espai lògic (= entrades de la TP)?
    - $2^{24}$  bytes totals /  $2^8$  bytes per pàgina =  $2^{16}$  pàgines
  - Dimensions de la TP =  $2^{16}$  entrades × (1 + 1 + **6**) bits =  **$2^{16}$  bytes**

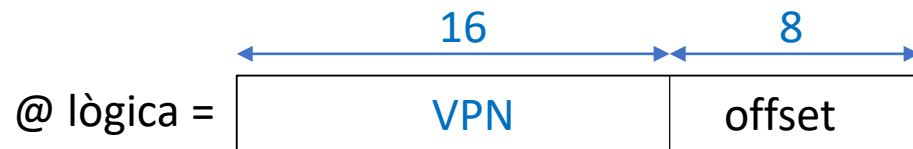
# Problema 7.1

- b)** A la taula següent apareix una llista d'adreces virtuals (en hexadecimal) que són generades pel processador. Omple la taula indicant, per a cadascuna de les adreces, el número de pàgina en hexadecimal corresponent (columna A), si l'accés produeix una fallada de pàgina o no (columna B), i, en cas que la fallada de pàgina hagi produït un reemplaçament, el número de pàgina reemplaçada (columna C). Considera que inicialment no hi ha cap pàgina carregada a memòria física.

Adreça (hex)	A Núm. pàgina (hex)	B Fallada de pàgina (SI o NO)	C Núm. pàgina reemplaçada (hex)
000023			
000101			
000102			
000200			
000010			
000111			
000416			
000520			
000101			

# Problema 7.1

- A: Determinar els números de pàgina (VPN)



Adreça (hex)	A – VPN (hex)	B – Fallada de pàgina (S/N)	C – VPN pàg. Reemplaçada (hex)
0000 23	0		
0001 01	1		
0001 02	1		
0002 00	2		
0000 10	0		
0001 11	1		
0004 16	4		
0005 20	5		
0001 01	1		

# Problema 7.1

- Afegim una columna amb els VPNs presents a MP (en ordre LRU)
- El programa pot usar 3 marcs i estan inicialment lliures

Cada programa pot tenir a memòria física un màxim de 3 pàgines simultàniament.

Considera que inicialment no hi ha cap pàgina carregada a memòria física.

Adreça (hex)	A – VPN (hex)	B – Fallada de pàgina (S/N)	C – VPN pàg. Reemplaçada (hex)	LRU: VPNs en MP (--, --. --)
0000 23	0			
0001 01	1			
0001 02	1			
0002 00	2			
0000 10	0			
0001 11	1			
0004 16	4			
0005 20	5			
0001 01	1			

# Problema 7.1

- 000023 → pàgina 0: **fallada**
- 000101 → pàgina 1: **fallada**
- 000102 → pàgina 1: **encert**
- 000200 → pàgina 2: **fallada**

Adreça (hex)	A – VPN (hex)	B – Fallada de pàgina (S/N)	C – VPN pàg. Reemplaçada (hex)	LRU: VPNs en MP (--, --. --)
0000 23	0	S		0, --, --
0001 01	1	S		0, 1, --
0001 02	1	N		0, 1, --
0002 00	2	S		0, 1, 2
0000 10	0			
0001 11	1			
0004 16	4			
0005 20	5			
0001 01	1			

# Problema 7.1

- 000010 → pàgina 0: **encert** → passa a ser la més recent
- 000111 → pàgina 1: **encert** → passa a ser la més recent

Adreça (hex)	A – VPN (hex)	B – Fallada de pàgina (S/N)	C – VPN pàg. Reemplaçada (hex)	LRU: VPNs en MP (--, --, --)
0000 23	0	S		0, --, --
0001 01	1	S		0, 1, --
0001 02	1	N		0, 1, --
0002 00	2	S		0, 1, 2
0000 10	0	N		1, 2, 0
0001 11	1	N		2, 0, 1
0004 16	4			
0005 20	5			
0001 01	1			



# Problema 7.1

- 000416 → pàgina 4: **fallada** → **reemplaça la 2**

Adreça (hex)	A – VPN (hex)	B – Fallada de pàgina (S/N)	C – VPN pàg. Reemplaçada (hex)	LRU: VPNs en MP (--, --. --)
0000 23	0	S		0, --, --
0001 01	1	S		0, 1, --
0001 02	1	N		0, 1, --
0002 00	2	S		0, 1, 2
0000 10	0	N		1, 2, 0
0001 11	1	N		<del>2</del> , 0, 1
0004 16	4	S	2	0, 1, 4
0005 20	5			
0001 01	1			

# Problema 7.1

- 000416 → pàgina 4: **fallada** → **reemplaça la 2**
- 000520 → pàgina 5: **fallada** → **reemplaça la 0**
- 000101 → pàgina 1: **encert** → **passa a ser la més recent**

Adreça (hex)	A – VPN (hex)	B – Fallada de pàgina (S/N)	C – VPN pàg. Reemplaçada (hex)	LRU: VPNs en MP (--, --, --)
0000 23	0	S		0, --, --
0001 01	1	S		0, 1, --
0001 02	1	N		0, 1, --
0002 00	2	S		0, 1, 2
0000 10	0	N		1, 2, 0
0001 11	1	N		2, 0, 1
0004 16	4	S	2	<del>0</del> , 1, 4
0005 20	5	S	0	1, 4, 5
0001 01	1	N		4, 5, 1

# Problema 7.2

- 7.2. Un sistema experimental es gestiona amb memòria virtual sota paginació amb algorisme de reemplaçament LRU. Aquest sistema té 64 KB de memòria virtual i 8 KB de memòria física. Les pàgines tenen 1 KB de mida. La taula de pàgines està carregada sempre a memòria física, ocupant l'últim marc de pàgina.

Suposem que una certa aplicació és ubicada dins l'espai lògic en les següents adreces:

- El codi del programa se situa en el rang d'adreces: 0x0000 - 0x03FF
- Les dades del programa se situen en el rang d'adreces: 0x0400 - 0x47FF

Dins les dades del programa hi ha una matriu que en alt nivell ha estat declarada com:

```
int MAT[32][256];
```

Aquesta matriu s'emmagatzema a memòria a partir de l'adreça base 0x0400. Tingues en compte que els enters es codifiquen en 16 bits. La resta de variables del programa, així com la pila, s'emmagatzemen a partir de l'adreça 0x4400.

Considerem el següent fragment del codi de l'aplicació:

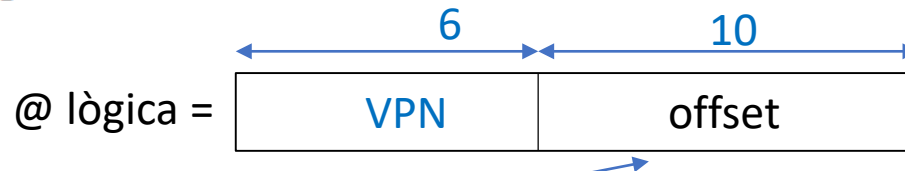
```
for (i=0; i<32; i++)
    for (j=0; j<256; j++)
        MAT[i][j] = i+j;
```

on:

- El compilador reserva dos registres del processador per a les variables  $i$  i  $j$ .
- La taula de pàgines abans de començar a executar l'anterior codi indica que a memòria física només tenim la pàgina corresponent al codi de l'aplicació.

# Problema 7.2

- a) Quins elements de MAT es porten a memòria física en la primera fallada de pàgina que es produeix?




- Pàgines de 1KB =  $2^{10}$  bytes
- El primer element accedit és  $M[0][0]$
- $@M[0][0] = 0x0400 = 0000\ 0100\ 0000\ 0000_2 \rightarrow$  pàgina VPN=1
- Una pàgina =  $2^{10}$  bytes / 2 bytes per element = 512 elements  
= 512 elem. / 256 elem. per fila = 2 files
- La primera fallada porta a MP **les files 0 i 1 de MAT**

# Problema 7.2

b) Indica, justificant-ho raonadament, el nombre total de fallades de pàgina que es produiran.

- Quines pàgines ocupa el codi?

- Adreces 0x0000 – 0x3FFF: és un rang de  $2^{10}$  bytes = 1 pàgina (VPN=0), però ja està carregada en memòria inicialment



- La taula de pàgines abans de començar a executar l'anterior codi indica que a memòria física només tenim la pàgina corresponent al codi de l'aplicació.

- Quines pàgines ocupa la matriu?

- A mitja pàgina per fila, ocupa 16 pàgines (pàgines VPN = 1 a 16)

- La matriu es recorre per files

- Es visiten seqüencialment totes els elements de cada pàgina abans de passar a la següent, i no es repeteix cap element
- Hi haurà 1 fallada de pàgina en el primer element de cada pàgina
- En total: **16 fallades**

# Problema 7.2

- c) En alguns llenguatges, com ara FORTRAN, els elements de les matrius es guarden en posicions consecutives però *per columnes* en lloc de *per files*. Considerant aquesta altra manera d'emmagatzemar les matrius, quins elements de MAT es portarien a memòria física en la primera fallada de pàgina?
- La primera fallada succeeix amb MAT[0][0], pàgina VPN=1
  - Una pàgina = 512 elements  
= 512 elem. / 32 elem. per columna = 16 columnes
  - La fallada porta a MP **les columnes 0 a 15 de MAT**

# Problema 7.2

- d) Considerant l'emmagatzematge per columnes de l'apartat anterior, indica, justificant-ho raonadament, el nombre total de fallades de pàgina que es produirien.
- Una pàgina ocupa 16 columnes, però recorrem MAT per files
  - Fila 0
    - Visitarem una nova pàgina cada 16 columnes
    - En total,  $256 \text{ columnes} / 16 \text{ columnes per pàgina} = 16 \text{ pàgines}$
    - La MP conté  $2^{13} \text{ bytes} / 2^{10} \text{ bytes per pàgina} = 8 \text{ marcs de pàgina}$
    - Disposem de 6 dels 8 marcs de MP
      - El marc 0 conté el codi, accedit sempre  $\rightarrow$  LRU no el reemplaça mai
      - El marc 7 està reservat a la TP
    - Al final de la fila sols conservem en MP les 6 últimes pàgines: 11 a 16
  - Fila 1
    - $\text{MAT}[1][0]$  pertany a la pàgina 1, però ja no està present en MP
    - Al llarg de la fila, tornem a visitar les 16 pàgines i fallem a totes
  - En total,  $16 \text{ fallades per fila} = 16 \times 32 = \mathbf{512 \text{ fallades}}$

# Problema 7.4

7.4. Tenim un processador amb memòria virtual basada en paginació. El sistema de memòria virtual té les següents característiques:

- 16 bits d'adreça lògica, i 15 bits d'adreça física
- mida de pàgines: 8 KB
- reemplaçament LRU

El contingut inicial de la taula de pàgines es mostra a continuació, on P és el bit de presència, D és el bit de pàgina modificada i PPN el número de pàgina física.

	P	D	PPN
0	0	0	-
1	0	0	-
2	1	0	0
3	1	0	1
4	1	0	2
5	0	0	-
6	0	0	-
7	0	0	-

A continuació també es mostra el contingut inicial de la memòria, en la qual la pàgina física 1 (lògica 3) és la que fa menys temps que ha estat accedida, la següent és la 0 (lògica 2) i la que fa més temps és la 2 (lògica 4).

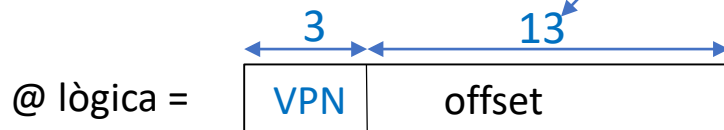
pàgina física	pàgina lògica
0	2
1	3
2	4
3	-



# Problema 7.4

- Calculem els VPN:

- 16 bits d'adreça lògica,
- mida de pàgines: 8 KB



0xF458 = 1111 0100 0101 1000

VPN = 7

La següent taula mostra una seqüència de referències a memòria (E: escriptura/ L: lectura). Emplena la taula indicant, per cada referència el número de pàgina lògica (VPN) i el número de pàgina física resultant de la traducció (PPN). Indica també amb una creu (X) si es produeix fallada de pàgina, si es llegeix del disc, i si s'escriu al disc. Indica també, en cas de reemplaçar una pàgina, el VPN i PPN de la pàgina reemplaçada. Per últim, indica també el contingut final de la taula de pàgines i de la memòria física.

adr. lògica (hex)	VPN (hex)	PPN (hex)	fallada de pàgina	lectura disc	escrip tura disc	pàg. reemplaçada	
						VPN (hex)	PPN (hex)
E F458							
E 8666							
L 1BBF							
E 5C44							
L 6600							
L 4000							

# Problema 7.4

- Omplim la columna dels VPNs
- Afegim 2 columnes més:
  - VPNs presents en MP (en ordre LRU): per saber quina pàgina reemplaçar (en negreta les pàgines Dirty)
  - VPNs presents en MP (en l'ordre dels marcs ocupats): per saber quin marc ocupava cada pàgina reemplaçada
- (amb les 2 columnes ja no caldria representar la TP) →

	P	D	PPN
0	0	0	
1	0	0	
2	1	0	0
3	1	0	1
4	1	0	2
5	0	0	
6	0	0	
7	0	0	

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escr. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7		si	si					
E	8666	4								
L	1BBF	0								
E	5C44	2								
L	6600	3								
L	4000	2								

# Problema 7.4

- Escriptura a VPN 7, P==0 → fallada
- Llegim pàgina al disc

	P	D	PPN
0	0	0	
1	0	0	
2	1	0	0
3	1	0	1
4	1	0	2
5	0	0	
6	0	0	
→ 7	0	0	

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escr. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7		si	si					
E	8666	4								
L	1BBF	0								
E	5C44	2								
L	6600	3								
L	4000	2								

# Problema 7.4

- Escriptura a VPN 7, P==0 → fallada
- Llegim pàgina al disc
- Ocupa el marc 3 que està lliure
- Actualitzem la TP: **P=1, D=1, PPN=3**

	P	D	PPN
0	0	0	
1	0	0	
2	1	0	0
3	1	0	1
4	1	0	2
5	0	0	
6	0	0	
→ 7	<b>1</b>	<b>1</b>	<b>3</b>

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escr. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	<b>3</b>	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4								
L	1BBF	0								
E	5C44	2								
L	6600	3								
L	4000	2								

# Problema 7.4

- Escriptura a VPN 4, P=1 → encert, PPN=2
- Passa a ser la més recent
- Posem el bit D=1

	P	D	PPN
0	0	0	
1	0	0	
2	1	0	0
3	1	0	1
→ 4	1	1	2
5	0	0	
6	0	0	
7	1	1	3

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escr. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	3	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4	2						2, 3, <b>7</b> , <b>4</b>	2, 3, <b>4</b> , <b>7</b>
L	1BBF	0								
E	5C44	2								
L	6600	3								
L	4000	2								

# Problema 7.4

- Lectura a VPN 0, P=0 → fallada
- Llegim pàgina del disc

	P	D	PPN
→ 0	0	0	
1	0	0	
2	1	0	0
3	1	0	1
4	1	1	2
5	0	0	
6	0	0	
7	1	1	3

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escr. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	3	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4	2						2, 3, <b>7</b> , <b>4</b>	2, 3, <b>4</b> , <b>7</b>
L	1BBF	0		si	si					
E	5C44	2								
L	6600	3								
L	4000	2								

# Problema 7.4

- Lectura a VPN 0, P=0 → fallada
- Llegim pàgina del disc
- Reemplacem pàgina 2
  - ocupava marc 0 i tenia D=0 → No cal escriure a disc

	P	D	PPN
→ 0	0	0	
1	0	0	
2	1	0	0
3	1	0	1
4	1	1	2
5	0	0	
6	0	0	
7	1	1	3

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escr. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	3	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4	2						<del>2</del> , 3, <b>7</b> , 4	<del>2</del> , 3, 4, <b>7</b>
L	1BBF	0	0	si	si		2	0	3, <b>7</b> , 4, <b>0</b>	<b>0</b> , 3, 4, <b>7</b>
E	5C44	2								
L	6600	3								
L	4000	2								

# Problema 7.4

- Lectura a VPN 0, P=0 → fallada
- Llegim pàgina del disc
- Reemplacem pàgina 2
  - ocupava marc 0 i tenia D=0 → No cal escriure a disc
- Actualitzem entrades 0 i 2 de la TP

	P	D	PPN
→ 0	1	0	0
1	0	0	
2	0	0	
3	1	0	1
4	1	1	2
5	0	0	
6	0	0	
7	1	1	3

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escr. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	3	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4	2						2, 3, <b>7</b> , <b>4</b>	2, 3, <b>4</b> , <b>7</b>
L	1BBF	0	0	si	si		2	0	3, <b>7</b> , <b>4</b> , <b>0</b>	<b>0</b> , 3, <b>4</b> , <b>7</b>
E	5C44	2								
L	6600	3								
L	4000	2								



# Problema 7.4

- Escriptura a VPN 2, P=0 → fallada
- Llegim pàgina del disc

	P	D	PPN
0	1	0	0
1	0	0	
→ 2	0	0	
3	1	0	1
4	1	1	2
5	0	0	
6	0	0	
7	1	1	3

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escr. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	3	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4	2						2, 3, <b>7</b> , 4	2, 3, <b>4</b> , 7
L	1BBF	0	0	si	si		2	0	3, <b>7</b> , 4, 0	0, 3, <b>4</b> , 7
E	5C44	2		si	si					
L	6600	3								
L	4000	2								

# Problema 7.4

- Escriptura a VPN 2, P=0 → fallada
- Llegim pàgina del disc
- Reemplacem pàgina 3
  - ocupava marc 1 i tenia D=0 → No cal escriure a disc

	P	D	PPN
0	1	0	0
1	0	0	
2	0	0	
3	1	0	1
4	1	1	2
5	0	0	
6	0	0	
7	1	1	3

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escr. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	3	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4	2						2, 3, <b>7</b> , <b>4</b>	2, 3, <b>4</b> , <b>7</b>
L	1BBF	0	0	si	si		2	0	<del>2</del> , <b>7</b> , <b>4</b> , 0	0, <del>2</del> , <b>4</b> , <b>7</b>
E	5C44	2	1	si	si		3	1	<b>7</b> , <b>4</b> , 0, <b>2</b>	0, <b>2</b> , <b>4</b> , <b>7</b>
L	6600	3								
L	4000	2								

# Problema 7.4

- Escriptura a VPN 2, P=0 → fallada
- Llegim pàgina del disc
- Reemplacem pàgina 3
  - ocupava marc 1 i tenia D=0 → No cal escriure a disc
- Actualitzem entrades 2 i 3 de la TP

	P	D	PPN
0	1	0	0
1	0	0	
→ 2	1	1	1
3	0	0	
4	1	1	2
5	0	0	
6	0	0	
7	1	1	3

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escr. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	3	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4	2						2, 3, <b>7</b> , <b>4</b>	2, 3, <b>4</b> , <b>7</b>
L	1BBF	0	0	si	si		2	0	3, <b>7</b> , <b>4</b> , 0	0, 3, <b>4</b> , <b>7</b>
E	5C44	2	1	si	si		3	1	<b>7</b> , <b>4</b> , 0, <b>2</b>	0, <b>2</b> , <b>4</b> , <b>7</b>
L	6600	3								
L	4000	2								

# Problema 7.4

- Lectura a VPN 3, P=0 → fallada
- Llegim pàgina del disc

	P	D	PPN
0	1	0	0
1	0	0	
2	1	1	1
→ 3	0	0	
4	1	1	2
5	0	0	
6	0	0	
7	1	1	3

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escr. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	3	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4	2						2, 3, <b>7</b> , 4	2, 3, <b>4</b> , 7
L	1BBF	0	0	si	si		2	0	3, <b>7</b> , 4, 0	0, 3, <b>4</b> , 7
E	5C44	2	1	si	si		3	1	<b>7</b> , 4, 0, 2	0, <b>2</b> , 4, 7
L	6600	3		Si	Si					
L	4000	2								

# Problema 7.4

- Lectura a VPN 3, P=0 → fallada
- Llegim pàgina del disc
- Reemplacem pàgina 7
  - ocupava marc 3 i tenia D=1 → Cal escriure-la a disc

	P	D	PPN
0	1	0	0
1	0	0	
2	1	1	1
3	0	0	
4	1	1	2
5	0	0	
6	0	0	
7	1	1	3

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escri. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	3	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4	2						2, 3, <b>7</b> , 4	2, 3, <b>4</b> , 7
L	1BBF	0	0	si	si		2	0	3, <b>7</b> , 4, 0	0, 3, <b>4</b> , 7
E	5C44	2	1	si	si		3	1	<del>7</del> , 4, 0, 2	0, 2, 4, <del>7</del>
L	6600	3	3	Si	Si	si	7	3	<b>4</b> , 0, 2, <b>3</b>	0, 2, 4, <b>3</b>
L	4000	2								

# Problema 7.4

- Lectura a VPN 3, P=0 → fallada
- Llegim pàgina del disc
- Reemplacem pàgina 7
  - ocupava marc 3 i tenia D=1 → Cal escriure-la a disc
- Actualitzem entrades 3 i 7 de la TP

	P	D	PPN
0	1	0	0
1	0	0	
2	1	1	1
3	1	0	3
4	1	1	2
5	0	0	
6	0	0	
7	0	0	

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escri. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	3	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4	2						2, 3, <b>7</b> , 4	2, 3, <b>4</b> , 7
L	1BBF	0	0	si	si		2	0	3, <b>7</b> , 4, 0	0, 3, <b>4</b> , 7
E	5C44	2	1	si	si		3	1	<b>7</b> , 4, 0, 2	0, <b>2</b> , 4, 7
L	6600	3	3	Si	Si	si	7	3	<b>4</b> , 0, 2, <b>3</b>	0, <b>2</b> , 4, <b>3</b>
L	4000	2								

# Problema 7.4

- Lectura a VPN 2, P=1 → encert (PPN=1)
- Passa a ser la més recent

	P	D	PPN
0	1	0	0
1	0	0	
→ 2	1	1	1
3	1	0	3
4	1	1	2
5	0	0	
6	0	0	
7	0	0	

@ lògica		VPN	PPN	fall. pàg.?	lect. disc?	escri. disc?	Pàg. reemplaçada		LRU de MP (VPN)	Estat de MP (VPN)
							VPN	PPN		
									4, 2, 3, --	2, 3, 4, --
E	F458	7	3	si	si				4, 2, 3, <b>7</b>	2, 3, 4, <b>7</b>
E	8666	4	2						2, 3, <b>7</b> , <b>4</b>	2, 3, <b>4</b> , <b>7</b>
L	1BBF	0	0	si	si		2	0	3, <b>7</b> , <b>4</b> , 0	0, 3, <b>4</b> , <b>7</b>
E	5C44	2	1	si	si		3	1	<b>7</b> , <b>4</b> , 0, <b>2</b>	0, <b>2</b> , <b>4</b> , <b>7</b>
L	6600	3	3	Si	Si	si	7	3	<b>4</b> , 0, <b>2</b> , 3	0, <b>2</b> , <b>4</b> , 3
L	4000	2	1						<b>4</b> , 0, 3, <b>2</b>	0, <b>2</b> , <b>4</b> , 3

# Problema 7.3

**7.3.** Considerem un computador amb processador MIPS com l'estudiat a classe, que té una memòria cache de dades amb les següents característiques:

- associativa per conjunts de 4 vies (és a dir, de 4 blocs per conjunt)
- 64 conjunts
- 32 bytes per bloc
- algorisme de reemplaçament LRU

Considerem també que té un TLB de dades amb les següents característiques:

- completament associatiu
- 32 entrades
- mida de pàgina: 8 KB
- algorisme de reemplaçament LRU

Sobre aquest sistema s'executen 2 versions diferents d'una mateixa aplicació:

```
int A[128][1024]; /* emmagatzemada a partir de l'adreça 0 */


/* versió A */
sumA = 0;
for (i=0; i<128; i++)
    for (j=0; j<1024; j++)
        sumA = sumA + A[i][j];
```

- Prescindirem de la part de cache (idèntica al problema 6.11, ja resolt)



# Problema 7.3

Considerem també que té un TLB de dades amb les següents característiques:

- completament associatiu
  - 32 entrades
  - mida de pàgina: 8 KB
  - algorisme de reemplaçament LRU
- 
- Quants elements conformen una pàgina?
    - 1 pàgina =  $2^{13}$  bytes =  $2^{13} / 4$  bytes per element = 2048 elements
  - Quantes pàgines ocupa la matriu?
    - 1 pàgina = 2048 elements = 2 files
    - En total, ocupa 128 files / 2 files per pàgina = **64 pàgines**
- 

# Problema 7.3

Considerem també que té un TLB de dades amb les següents característiques:

- completament associatiu
- 32 entrades
- mida de pàgina: 8 KB
- algorisme de reemplaçament LRU

Suposant que les variables  $i$ ,  $j$ , i  $sumA$  s'emmagatzemen en registres, indica quantes fallades hi ha            al TLB,           

- La versió A del bucle recorre la matriu **per files**
  - Recorre tots els elements de cada pàgina abans d'avançar a la següent, i després ja no accedeix més a la pàgina recorreguda
  - El TLB falla en el 1er element de cada pàgina i porta la PTE al TLB
- En total, recorre 64 pàgines → fa **64 fallades de TLB**

# Problema 7.3

```
/* versió B */  
sumA = 0;  
for (j=0; j<1024; j++)  
    for (i=0; i<128; i++)  
        sumA = sumA + A[i][j];
```

- La versió B del bucle recorre la matriu **per columnes**
- A la columna 0
  - El TLB falla a la fila 0, i encerta a la fila 1 (pàgines de 2 files)
  - Falla de nou a la fila 2 i encerta a la fila 3
  - Etc.
  - Un cop visitades 64 files ja ha omplert les 32 entrades del TLB
  - A partir de la fila 64 comença a reemplaçar les entrades del TLB
- A la columna 1
  - Repeteix pàgines ja visitades però les traduccions han estat reemplaçades
  - Falla 1 de cada 2 elements de la columna
  - Etc.
- En total, falla en un de cada 2 accessos
  - $1/2 \times 128 \times 1024 = \mathbf{65536 \text{ fallades}}$

# Problema 7.5

7.5. Considerem el següent codi escrit en ensamblador del MIPS, en què a l'etiqueta A li correspon l'adreça 0x10010000:

```
    la    $s0, A
    li    $s1, 0
    li    $s2, 512000
for: bge   $s1, $s2, fi
    sll   $t0, $s1, 2
    addu  $t0, $s0, $t0
    lw    $t1, 0($t0)
    lw    $t2, 8192($t0)
    addu  $t2, $t2, $t1
    sw    $t2, 8192($t0)
    sw    $t2, 16384($t0)
    addiu $s1, $s1, 512
    b     for
fi:
```

@A és múltiple de la mida de pàgina → el primer element està a l'inici d'una pàgina

Suposant que el sistema de memòria té les pàgines de mida 8 KB i que utilitza un TLB de 4 entrades (completament associatiu, i amb reemplaçament LRU), respon les següents preguntes:

- Per a cadascun dels accessos a dades de memòria (instruccions en negreta) i per a les primeres 5 iteracions del bucle, indica a quina pàgina de la memòria virtual s'està accedint.
- Indica la quantitat de fallades de TLB en tot el bucle.
- Respon a les mateixes preguntes anteriors, suposant ara que el sistema de memòria té les pàgines de mida 4 KB.

# Problema 7.5: una possible estratègia

```
la    $s0, A
li    $s1, 0
li    $s2, 512000
for:  bge    $s1, $s2, fi
      sll    $t0, $s1, 2
      addu   $t0, $s0, $t0
      lw     $t1, 0($t0)
      lw     $t2, 8192($t0)
      addu   $t2, $t2, $t1
      sw     $t2, 8192($t0)
      sw     $t2, 16384($t0)
      addiu  $s1, $s1, 512
      b      for
fi:
```

- Podem expressar el codi en C?

- \$s1 és el comptador del bucle
- Compta de 0 a 512000
- Incrementa en 512 per iteració

# Problema 7.5: una possible estratègia

```
la      $s0, A
li      $s1, 0
li      $s2, 512000
for: bge $s1, $s2, fi
    sll  $t0, $s1, 2
    addu $t0, $s0, $t0
    lw   $t1, 0($t0)
    lw   $t2, 8192($t0)
    addu $t2, $t2, $t1
    sw   $t2, 8192($t0)
    sw   $t2, 16384($t0)
    addiu $s1, $s1, 512
    b    for
fi:
```

- Podem expressar el codi en C?
  - \$s1 és el comptador del bucle
  - Compta de 0 a 512000
  - Incrementa en 512 per iteració
  - **\$s1 indexa l'estructura de dades A**
  - **Els elements són de 4 bytes**

# Problema 7.5: una possible estratègia

```
la      $s0, A
li      $s1, 0
li      $s2, 512000
for: bge $s1, $s2, fi
sll     $t0, $s1, 2
addu    $t0, $s0, $t0
lw      $t1, 0($t0)
lw      $t2, 8192($t0)
addu    $t2, $t2, $t1
sw      $t2, 8192($t0)
sw      $t2, 16384($t0)
addiu   $s1, $s1, 512
b       for
fi:
```

- Podem expressar el codi en C?
  - \$s1 és el comptador del bucle
  - Compta de 0 a 512000
  - Incrementa en 512 per iteració
  - \$s1 indexa l'estructura de dades A
  - Els elements són de 4 bytes
  - En cada iteració l'índex "s1" "avança" 512 elements
  - Podem imaginar que A és una matriu de 512 columnes, i que recorrem la columna 0, fila per fila!

# Problema 7.5: una possible estratègia

```
la      $s0, A
li      $s1, 0
li      $s2, 512000
for: bge $s1, $s2, fi
sll     $t0, $s1, 2
addu    $t0, $s0, $t0


|      |                   |
|------|-------------------|
| lw   | \$t1, 0(\$t0)     |
| lw   | \$t2, 8192(\$t0)  |
| addu | \$t2, \$t2, \$t1  |
| sw   | \$t2, 8192(\$t0)  |
| sw   | \$t2, 16384(\$t0) |


addiu   $s1, $s1, 512
b       for
fi:
```

- Podem expressar el codi en C?
  - \$s1 és el comptador del bucle
  - Compta de 0 a 512000
  - Incrementa en 512 per iteració
  - \$s1 indexa l'estructura de dades A
  - Els elements són de 4 bytes
  - En cada iteració l'índex "s1" "avança" 512 elements
  - Podem imaginar que A és una matriu de 512 columnes, i que recorrem la columna 0, fila per fila!
  - El primer lw accedeix a  $A[s1][0]$
  - El segon lw i el primer sw accedeixen 8192 bytes més endavant = 4 files →  $A[s1+4][0]$
  - El segon sw accedeix 16384 bytes més endavant = 8 files →  $A[s1+8][0]$



# Problema 7.5: una possible estratègia

```
la      $s0, A
li      $s1, 0
li      $s2, 512000
for: bge $s1, $s2, fi
sll     $t0, $s1, 2
addu    $t0, $s0, $t0
lw      $t1, 0($t0)
lw      $t2, 8192($t0)
addu    $t2, $t2, $t1
sw      $t2, 8192($t0)
sw      $t2, 16384($t0)
addiu   $s1, $s1, 512
b       for
fi:
```

```
int A[2000][512], s1, t2;
for (s1=0; s1<1000; s1++)
{
    t2 = A[s1][0] + A[s1+4][0];
    A[s1+4] = t2;
    A[s1+8] = t2;
}
```

- Podem expressar el codi en C?
  - \$s1 és el comptador del bucle
  - Compta de 0 a 512000
  - Incrementa en 512 per iteració
  - \$s1 indexa l'estructura de dades A
  - Els elements són de 4 bytes
  - En cada iteració l'índex "s1" "avança" 512 elements
  - Podem imaginar que A és una matriu de 512 columnes, i que recorrem la columna 0, fila per fila!
  - El primer lw accedeix a A[s1][0]
  - El segon lw i el primer sw accedeixen 8192 bytes més endavant = 4 files → A[s1+4][0]
  - El segon sw accedeix 16384 bytes més endavant = 8 files → A[s1+8][0]

• Programa equivalent, en C

# Problema 7.5

Suposant que el sistema de memòria té les pàgines de mida 8 KB i que utilitza un TLB de 4 entrades (completament associatiu, i amb reemplaçament LRU), respon les següents preguntes:

- a) Per a cadascun dels accessos a dades de memòria (instruccions en negreta) i per a les primeres 5 iteracions del bucle, indica a quina pàgina de la memòria virtual s'està accedint.
- Una pàgina conté
    - 8KB / 4 bytes per element = 2048 elements = 4 files
  - A la iteració 0, accedim a les files 0, 4 i 8 → pàgines 0, 1 i 2

S1	lw A[s1][0]	lw A[s1+4][0]	sw A[s1+4][0]	sw A[s1+8][0]
0	0	1	1	2
1				
2				
3				
4				

# Problema 7.5

Suposant que el sistema de memòria té les pàgines de mida 8 KB i que utilitza un TLB de 4 entrades (completament associatiu, i amb reemplaçament LRU), respon les següents preguntes:

- a) Per a cadascun dels accessos a dades de memòria (instruccions en negreta) i per a les primeres 5 iteracions del bucle, indica a quina pàgina de la memòria virtual s'està accedint.
- Per cada iteració avancem 1 fila de la matriu
    - Calen 4 iteracions per avançar 1 pàgina

S1	lw A[s1][0]	lw A[s1+4][0]	sw A[s1+4][0]	sw A[s1+8][0]
0	0	1	1	2
1	0	1	1	2
2	0	1	1	2
3	0	1	1	2
4	1	2	2	3

# Problema 7.5

Suposant que el sistema de memòria té les pàgines de mida 8 KB i que utilitza un TLB de 4 entrades (completament associatiu, i amb reemplaçament LRU), respon les següents preguntes:

b) Indica la quantitat de fallades de TLB en tot el bucle.

s1	lw	lw/sw	sw
0	0	1	2
4	1	2	3
8	2	3	4
12	3	4	5
...	...	...	...

- Sabem d'entrada quines pàgines es visiten en cada iteració i en cada accés, i omplim la taula



# Problema 7.5

Suposant que el sistema de memòria té les pàgines de mida 8 KB i que utilitza un TLB de 4 entrades (completament associatiu, i amb reemplaçament LRU), respon les següents preguntes:

b) Indica la quantitat de fallades de TLB en tot el bucle.

s1	lw	lw/sw	sw
0	0	1	2
4	1	2	3
8	2	3	4
12	3	4	5
...	...	...	...

- Fem un recorregut de les pàgines, observant les fallades de TLB
- En les files 0 a 7: fallem les pàgines 0, 1, 2 i 3 (TLB queda pla)
  - LRU: 0, 1, 2, 3 en aquest ordre

# Problema 7.5

Suposant que el sistema de memòria té les pàgines de mida 8 KB i que utilitza un TLB de 4 entrades (completament associatiu, i amb reemplaçament LRU), respon les següents preguntes:

b) Indica la quantitat de fallades de TLB en tot el bucle.

s1	lw	lw/sw	sw
0	0	1	2
4	1	2	3
8	2	3	4
12	3	4	5
...	...	...	...

- Fem un recorregut de les pàgines, observant les fallades de TLB
- En les files 0 a 7: fallem les pàgines 0, 1, 2 i 3 (TLB queda pla)
  - LRU: ~~0~~, 1, 2, 3 en aquest ordre
- En les files 8 a 11: fallem a la pàgina 4, i reemplacem la pàgina 0
  - LRU: 1, 2, 3, 4

# Problema 7.5

Suposant que el sistema de memòria té les pàgines de mida 8 KB i que utilitza un TLB de 4 entrades (completament associatiu, i amb reemplaçament LRU), respon les següents preguntes:

b) Indica la quantitat de fallades de TLB en tot el bucle.

s1	lw	lw/sw	sw
0	0	1	2
4	1	2	3
8	2	3	4
12	3	4	5
...	...	...	...

- Fem un recorregut de les pàgines, observant les fallades de TLB
- En les files 0 a 7: fallem les pàgines 0, 1, 2 i 3 (TLB queda pla)
  - LRU: 0, 1, 2, 3 en aquest ordre
- En les files 8 a 11: fallem a la pàgina 4, i reemplacem la pàgina 0
  - LRU: ~~1~~, 2, 3, 4
- En les files 12 a 15: fallem a la pàgina 5, i reemplacem la pàgina 1
  - LRU: 2, 3, 4, 5

# Problema 7.5

Suposant que el sistema de memòria té les pàgines de mida 8 KB i que utilitza un TLB de 4 entrades (completament associatiu, i amb reemplaçament LRU), respon les següents preguntes:

b) Indica la quantitat de fallades de TLB en tot el bucle.

s1	lw	lw/sw	sw
0	0	1	2
4	1	2	3
8	2	3	4
12	3	4	5
16	4	5	6

- Fem un recorregut de les pàgines, observant les fallades de TLB
  - En les files 0 a 7: fallem les pàgines 0, 1, 2 i 3 (TLB queda pla)
    - LRU: 0, 1, 2, 3 en aquest ordre
  - En les files 8 a 11: fallem a la pàgina 4, i reemplacem la pàgina 0
    - LRU: 1, 2, 3, 4
  - En les files 12 a 15: fallem a la pàgina 5, i reemplacem la pàgina 1
    - LRU: 2, 3, 4, 5
- 
- En resum: Fallem 1 cop per cada 4 iteracions + 2 cops addicionals a la fila 0
  - En total:  $2 + 1000 \text{ iteracions} / 4 = \mathbf{252 \text{ fallades}}$



# Problema 7.5

c) Respon a les mateixes preguntes anteriors, suposant ara que el sistema de memòria té les pàgines de mida 4 KB.

- Una pàgina conté
  - $4\text{KB} / 4 \text{ bytes per element} = 1024 \text{ elements} = 2 \text{ files}$
- A la iteració  $s1=0$ , accedim a les files 0, 4 i 8  $\rightarrow$  pàgines 0, 2 i 4

S1	lw $A[s1][0]$	lw $A[s1+4][0]$	sw $A[s1+4][0]$	sw $A[s1+8][0]$
0	0	2	2	4
1				
2				
3				
4				

# Problema 7.5

- c) Respon a les mateixes preguntes anteriors, suposant ara que el sistema de memòria té les pàgines de mida 4 KB.
- Per cada iteració avancem 1 fila de la matriu
    - Calen 2 iteracions per avançar 1 pàgina

S1	lw A[s1][0]	lw A[s1+4][0]	sw A[s1+4][0]	sw A[s1+8][0]
0	0	2	2	4
1	0	2	2	4
2	1	3	3	5
3	1	3	3	5
4	2	4	4	6

# Problema 7.5

b) Indica la quantitat de fallades de TLB en tot el bucle.

s1	lw	lw/sw	sw
0	0	2	4
2	1	3	5
4	2	4	6
6	3	5	7
...	...	...	...

- Fem un recorregut de les pàgines, observant les fallades de TLB
- Fila 0: fallem les pàgines 0, 2, 4
- Fila 2: fallem la pàgina 1 (TLB ple)
  - LRU: 0, 2, 4, 1 en aquest ordre

# Problema 7.5

b) Indica la quantitat de fallades de TLB en tot el bucle.

s1	lw	lw/sw	sw
0	0	2	4
2	1	3	5
4	2	4	6
6	3	5	7
...	...	...	...

- Fem un recorregut de les pàgines, observant les fallades de TLB
- Fila 0: fallem les pàgines 0, 2, 4
- Fila 2: fallem la pàgina 1 (TLB ple)
  - LRU: ~~0~~, 2, 4, 1 en aquest ordre
  - Fallem la pàgina 3, reemplaça la 0
  - LRU: ~~2~~, 4, 1, 3
  - Fallem la pàgina 5, reemplaça la 2
  - LRU: 4, 1, 3, 5

# Problema 7.5

b) Indica la quantitat de fallades de TLB en tot el bucle.

s1	lw	lw/sw	sw
0	0	2	4
2	1	3	5
4	2	4	6
6	3	5	7
...	...	...	...

- Fem un recorregut de les pàgines, observant les fallades de TLB
- Fila 0: fallem les pàgines 0, 2, 4
- Fila 2: fallem la pàgina 1 (TLB ple)
  - LRU: 0, 2, 4, 1 en aquest ordre
  - Fallem la pàgina 3, reemplaça la 0
  - LRU: 2, 4, 1, 3
  - Fallem la pàgina 5, reemplaça la 2
  - LRU: ~~4~~, ~~1~~, ~~3~~, 5
- Fila 4: fallem les pàgines 2, 4 i 6
  - I reemplaçem les pàgines 4, 1, 3
  - LRU: 5, 2, 4, 6
- etc.

# Problema 7.5

b) Indica la quantitat de fallades de TLB en tot el bucle.

s1	lw	lw/sw	sw
0	0	2	4
2	1	3	5
4	2	4	6
6	3	5	7
...	...	...	...

- Fem un recorregut de les pàgines, observant les fallades de TLB
- Fila 0: fallem les pàgines 0, 2, 4
- Fila 2: fallem la pàgina 1 (TLB ple)
  - LRU: 0, 2, 4, 1 en aquest ordre
  - Fallem la pàgina 3, reemplaça la 0
  - LRU: 2, 4, 1, 3
  - Fallem la pàgina 5, reemplaça la 2
  - LRU: 4, 1, 3, 5
- Fila 4: fallem les pàgines 2, 4 i 6
  - I reemplaçem les pàgines 4, 1, 3
  - LRU: 5, 2, 4, 6
- etc.

En resum:

Fallem 3 cops per cada 2 iteracions

En total:  $3 \times 1000 \text{ iteracions} / 2 = \mathbf{1500 \text{ fallades}}$

# Problema 7.6

Considerem un sistema amb memòria virtual que té pàgines de 4 KB. A més, té un TLB de 4 entrades completament associatiu, on s'utilitza un reemplaçament LRU.

Considerem els següents continguts inicials del TLB (per al funcionament de l'algorisme LRU, considerem que les tres entrades vàlides han estat recentment accedides en el mateix ordre que ocupen dins el TLB):

TLB		
V	VPN	PPN
1	11	12
1	7	4
1	3	6
0	9	0

Considerem també els següents continguts inicials de la taula de pàgines. Per a noves pàgines assignades a MP, s'assignaran números de pàgina física (PPN) correlatius, a partir del número major (és a dir, els números 13, 14, etc.).

TP	
P	PPN
0	1
1	0
2	0
3	1
4	1
5	1
6	0
7	1
8	0
9	0
10	1
11	1

a) Per a cada una de les dues llistes de referències d'adreces virtuals que es mostren a continuació, indica quin serà l'estat final del TLB i de la taula de pàgines (TP). A més, per a cada referència, indica si es produeix una fallada al TLB, i si es produeix una fallada de pàgina.

- Llista 1: 4095, 31272, 15789, 15000, 7193, 4096, 8912
- Llista 2: 9452, 30964, 19136, 46502, 38110, 17653, 47480

# Problema 7.6

- **Llista 1:** Calculem els VPN = adreça / mida\_de\_pàgina
  - @ = 4095 → VPN = 4095 / 4096 = 0
- Columnes addicionals: VPNs del TLB (en ordre d'entrades i en ordre LRU)

@ (dec)	VPN	TLB			Fallada pàgina?	PPN
		Fallada TLB?	VPNs	VPNs (LRU)		
			11,7,3,--	11,7,3,--		
4095	0					
31272	7					
15789	3					
15000	3					
7193	1					
4096	1					
8912	2					



# Problema 7.6

- Podem omplir la columna TLB independentment de les 2 últimes
- VPN 0: **falla** al TLB → llegim PTE de la TP: **ocupa l'entrada lliure**

@ (dec)	VPN	TLB			Fallada pàgina?	PPN
		Fallada TLB?	VPNs	VPNs (LRU)		
			11,7,3,--	11,7,3,--		
4095	0	<b>si</b>	11,7,3, <b>0</b>	11,7,3, <b>0</b>		
31272	7					
15789	3					
15000	3					
7193	1					
4096	1					
8912	2					

# Problema 7.6

- VPN 7: **encert** al TLB → passa a ser la més recent
- VPN 3: **encert** al TLB → passa a ser la més recent
- VPN 3: **encert** al TLB

@ (dec)	VPN	TLB			Fallada pàgina?	PPN
		Fallada TLB?	VPNs	VPNs (LRU)		
			11,7,3,--	11,7,3,--		
4095	0	si	11,7,3,0	11,7,3,0		
31272	7	no	11, <b>7</b> ,3,0	11,3,0, <b>7</b>		
15789	3	no	11,7, <b>3</b> ,0	11,0,7, <b>3</b>		
15000	3	no	11,7, <b>3</b> ,0	11,0,7, <b>3</b>		
7193	1					
4096	1					
8912	2					

# Problema 7.6

- VPN 1: **fallada** al TLB → llegim PTE de la TP i **reemplaça VPN 11**
- VPN 1: **encert**

@ (dec)	VPN	TLB			Fallada pàgina?	PPN
		Fallada TLB?	VPNs	VPNs (LRU)		
			11,7,3,--	11,7,3,--		
4095	0	si	11,7,3,0	11,7,3,0		
31272	7	no	11,7,3,0	11,3,0,7		
15789	3	no	11,7,3,0	11,0,7,3		
15000	3	no	<del>11,7,3,0</del>	<del>11,0,7,3</del>		
7193	1	si	1,7,3,0	0,7,3,1		
4096	1	no	1,7,3,0	0,7,3,1		
8912	2					

# Problema 7.6

- VPN 2: **fallada** al TLB → llegim PTE de la TP i **reemplaça VPN 0**

@ (dec)	VPN	TLB			Fallada pàgina?	PPN
		Fallada TLB?	VPNs	VPNs (LRU)		
			11,7,3,--	11,7,3,--		
4095	0	si	11,7,3,0	11,7,3,0		
31272	7	no	11,7,3,0	11,3,0,7		
15789	3	no	11,7,3,0	11,0,7,3		
15000	3	no	11,7,3,0	11,0,7,3		
7193	1	si	1,7,3,0	0,7,3,1		
4096	1	no	1,7,3,0	0,7,3,1		
8912	2	si	1,7,3,2	7,3,1,2		

# Problema 7.6

- Ara repassem les fallades de pàgina, comprovant la TP
- VPN 0 → encert, PPN = 5
- VPN 7 → encert, PPN = 4

@ (dec)	VPN	Fallada pàgina?	PPN
4095	0	no	5
31272	7	no	4
15789	3		
15000	3		
7193	1		
4096	1		
8912	2		

TP		
	P	PPN
→ 0	1	5
1	0	
2	0	
3	1	6
4	1	9
5	1	11
6	0	
→ 7	1	4
8	0	
9	0	
10	1	3
11	1	12

# Problema 7.6

- VPN 3 → **encert**, PPN= 6
- VPN 1 → **fallada!**

@ (dec)	VPN	Fallada pàgina?	PPN
4095	0	no	5
31272	7	no	4
15789	3	no	6
15000	3	no	6
7193	1	si	
4096	1		
8912	2		

TP		
	P	PPN
0	1	5
→ 1	0	
2	0	
→ 3	1	6
4	1	9
5	1	11
6	0	
7	1	4
8	0	
9	0	
10	1	3
11	1	12

# Problema 7.6

- VPN 1 → **fallada**, la copiem en un marc lliure de MP: **PPN=13**
- VPN 1 → **encert**

@ (dec)	VPN	Fallada pàgina?	PPN
4095	0	no	5
31272	7	no	4
15789	3	no	6
15000	3	no	6
7193	1	<b>si</b>	<b>13</b>
4096	1	<b>no</b>	<b>13</b>
8912	2		

TP		
	P	PPN
0	1	5
1	<b>1</b>	<b>13</b>
2	0	
3	1	6
4	1	9
5	1	11
6	0	
7	1	4
8	0	
9	0	
10	1	3
11	1	12

# Problema 7.6

- VPN 2 → fallada!

@ (dec)	VPN	Fallada pàgina?	PPN
4095	0	no	5
31272	7	no	4
15789	3	no	6
15000	3	no	6
7193	1	si	13
4096	1	no	13
8912	2	si	

TP		
	P	PPN
0	1	5
1	1	13
2	0	
3	1	6
4	1	9
5	1	11
6	0	
7	1	4
8	0	
9	0	
10	1	3
11	1	12



# Problema 7.6

- VPN 2 → **fallada**, la copiem en un marc lliure de MP: **PPN=14**

@ (dec)	VPN	Fallada pàgina?	PPN
4095	0	no	5
31272	7	no	4
15789	3	no	6
15000	3	no	6
7193	1	si	13
4096	1	no	13
8912	2	si	14

TP		
	P	PPN
0	1	5
1	1	13
→ 2	1	14
3	1	6
4	1	9
5	1	11
6	0	
7	1	4
8	0	
9	0	
10	1	3
11	1	12

# Problema 7.6

- **Llista 1:** Estat final de la TP i del TLB

TP			TLB		
	P	PPN	V	VPN	PPN
0	1	5	1	1	13
1	1	13	1	7	4
2	1	14	1	3	6
3	1	6	1	2	14
4	1	9			
5	1	11			
6	0				
7	1	4			
8	0				
9	0				
10	1	3			
11	1	12			

# Problema 7.6

- **Llista 2:** procedim com abans, calculant els VPN
- I a continuació comprovant el TLB pas a pas
- VPN 2 → **fallada**: copiem la PTE a l'entrada lliure

@ (dec)	VPN	TLB			Fallada pàgina?	PPN
		Fallada TLB?	VPNs	VPNs (LRU)		
			11,7,3,--	11,7,3,--		
9452	2	si	11,7,3,2	11,7,3,2		
30964	7					
19136	4					
46502	11					
38110	9					
17653	4					
47480	11					

# Problema 7.6

- VPN 7 → **encert**: passa a ser la més recent
- VPN 4 → **fallada**: copiem la PTE **reemplaçant la VPN 11**

@ (dec)	VPN	TLB			Fallada pàgina?	PPN
		Fallada TLB?	VPNs	VPNs (LRU)		
			11,7,3,--	11,7,3,--		
9452	2	si	11,7,3,2	11,7,3,2		
30964	7	no	<del>11</del> ,7,3,2	<del>11</del> ,3,2,7		
19136	4	si	4,7,3,2	3,2,7,4		
46502	11					
38110	9					
17653	4					
47480	11					

# Problema 7.6

- VPN 11 → fallada: reemplaça la VPN 3

@ (dec)	VPN	TLB			Fallada pàgina?	PPN
		Fallada TLB?	VPNs	VPNs (LRU)		
			11,7,3,--	11,7,3,--		
9452	2	si	11,7,3,2	11,7,3,2		
30964	7	no	11,7,3,2	11,3,2,7		
19136	4	si	4,7, <del>3</del> ,2	<del>3</del> ,2,7,4		
46502	11	si	4,7, <b>11</b> ,2	2,7,4, <b>11</b>		
38110	9					
17653	4					
47480	11					

# Problema 7.6

- VPN 9 → **fallada: reemplaça la VPN 2**

@ (dec)	VPN	TLB			Fallada pàgina?	PPN
		Fallada TLB?	VPNs	VPNs (LRU)		
			11,7,3,--	11,7,3,--		
9452	2	si	11,7,3,2	11,7,3,2		
30964	7	no	11,7,3,2	11,3,2,7		
19136	4	si	4,7,3,2	3,2,7,4		
46502	11	si	4,7,11, <del>2</del>	<del>2</del> ,7,4,11		
38110	9	si	4,7,11, <b>9</b>	7,4,11, <b>9</b>		
17653	4					
47480	11					

# Problema 7.6

- VPN 4 → **encert**: passa a ser la més recent
- VPN 11 → **encert**: passa a ser la més recent

@ (dec)	VPN	TLB			Fallada pàgina?	PPN
		Fallada TLB?	VPNs	VPNs (LRU)		
			11,7,3,--	11,7,3,--		
9452	2	si	11,7,3,2	11,7,3,2		
30964	7	no	11,7,3,2	11,3,2,7		
19136	4	si	4,7,3,2	3,2,7,4		
46502	11	si	4,7,11,2	2,7,4,11		
38110	9	si	4,7,11,9	7,4,11,9		
17653	4	no	4,7,11,9	7,11,9,4		
47480	11	no	4,7,11,9	7,9,4,11		

# Problema 7.6

- Ara repassem les fallades de pàgina, comprovant la TP
- VPN 2 → **fallada!**

@ (dec)	VPN	Fallada pàgina?	PPN
9452	2	si	
30964	7		
19136	4		
46502	11		
38110	9		
17653	4		
47480	11		

TP		
	P	PPN
0	1	5
1	0	
2	0	
3	1	6
4	1	9
5	1	11
6	0	
7	1	4
8	0	
9	0	
10	1	3
11	1	12



# Problema 7.6

- Ara repassem les fallades de pàgina, comprovant la TP
- VPN 2 → **fallada**, la copiem en un marc lliure de MP: **PPN=13**

@ (dec)	VPN	Fallada pàgina?	PPN
9452	2	si	13
30964	7		
19136	4		
46502	11		
38110	9		
17653	4		
47480	11		

TP		
	P	PPN
0	1	5
1	0	
→ 2	1	13
3	1	6
4	1	9
5	1	11
6	0	
7	1	4
8	0	
9	0	
10	1	3
11	1	12

# Problema 7.6

- VPN 7 → encert: PPN=4
- VPN 4 → encert: PPN=9
- VPN 11 → encert: PPN=12

@ (dec)	VPN	Fallada pàgina?	PPN
9452	2	si	13
30964	7	no	4
19136	4	no	9
46502	11	no	12
38110	9		
17653	4		
47480	11		

TP		
	P	PPN
0	1	5
1	0	
2	1	13
3	1	6
→ 4	1	9
5	1	11
6	0	
→ 7	1	4
8	0	
9	0	
10	1	3
→ 11	1	12

# Problema 7.6

- VPN 9: **fallada!**

@ (dec)	VPN	Fallada pàgina?	PPN
9452	2	si	13
30964	7	no	4
19136	4	no	9
46502	11	no	12
38110	9	si	
17653	4		
47480	11		

TP		
	P	PPN
0	1	5
1	0	
2	1	13
3	1	6
4	1	9
5	1	11
6	0	
7	1	4
8	0	
9	0	
10	1	3
11	1	12



# Problema 7.6

- VPN 9: **fallada**, la copiem al següent marc lliure de MP: **PPN=14**

@ (dec)	VPN	Fallada pàgina?	PPN
9452	2	si	13
30964	7	no	4
19136	4	no	9
46502	11	no	12
38110	9	si	14
17653	4		
47480	11		

TP		
	P	PPN
0	1	5
1	0	
2	1	13
3	1	6
4	1	9
5	1	11
6	0	
7	1	4
8	0	
9	1	14
10	1	3
11	1	12



# Problema 7.6

- VPN 4 → encert: PPN=9
- VPN 11 → encert: PPN=12

@ (dec)	VPN	Fallada pàgina?	PPN
9452	2	si	13
30964	7	no	4
19136	4	no	9
46502	11	no	12
38110	9	si	14
17653	4	no	9
47480	11	no	12

TP		
	P	PPN
0	1	5
1	0	
2	1	13
3	1	6
→ 4	1	9
5	1	11
6	0	
7	1	4
8	0	
9	1	14
10	1	3
→ 11	1	12

# Problema 7.6

- **Llista 2:** Estat final de la TP i del TLB

TP		
	P	PPN
0	1	5
1	0	
2	1	13
3	1	6
4	1	9
5	1	11
6	0	
7	1	4
8	0	
9	1	14
10	1	3
11	1	12

TLB		
V	VPN	PPN
1	4	9
1	7	4
1	11	12
1	9	14

# Problema 7.6

- c) Considerant els paràmetres següents de dos sistemes que gestionen memòria virtual, indica quina serà la mida de la taula de pàgines en cada cas.

Mida adreces virtuals	Mida pàgina	Mida entrades TP
32 bits	4 KB	4 bytes
64 bits	16 KB	8 bytes

Adreces de 32 bits: Mida de pàgina = 4KB =  $2^{12}$  bytes

→ calen 12 bits per a l'offset

→ el VPN consta de  $32 - 12 = 20$  bits

→ La TP té  $2^{20}$  entrades → Mida total =  $2^{20} \times 4 \text{ bytes} = \mathbf{2^{24} \text{ bytes}}$

Adreces de 64 bits: Mida de pàgina = 16KB =  $2^{14}$  bytes

→ calen 14 bits per a l'offset

→ el VPN consta de  $64 - 14 = 50$  bits

→ La TP té  $2^{50}$  entrades → Mida total =  $2^{50} \times 8 \text{ bytes} = \mathbf{2^{53} \text{ bytes}}$

# Examen 2010/11-Q1 (adaptat)

Donat el següent programa en C

```
int M[144][256], i, tmp;
for (i=0; i< 128; i++) {
    tmp = M[i][0];
    M[i+4][0] = tmp;
    M[i+16][0] = M[i+16][0] + tmp;
}
```

S'ha compilat per a un processador de 32 bits amb memòria virtual paginada

- Les variables *i* i *tmp* s'han guardat en registres
- L'adreça lògica inicial de *M* és 0x00000000
- La mida de les pàgines és 4KB
- El TLB té 4 entrades (associatiu, amb reemplaçament LRU)

a) Per a cada un dels 4 accessos a la matriu *M*, indica a quina pàgina lògica (VPN) s'accedeix en cada una de les 16 primeres iteracions

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
M[i][0]																
M[i+4][0]																
M[i+16][0]																
M[i+16][0]																



# Examen 2010/11-Q1 (adaptat)

- Quants elements conté una pàgina?
  - $4\text{KB} = 2^{12} \text{ bytes} = 2^{12} / 4 \text{ bytes per element} = 2^{10} \text{ elements} = 4 \text{ files}$
  - A la fila 0 comença la pàgina 0
  - A la fila 4 comença la pàgina 1
  - A la fila 16 comença la pàgina 4

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
M[i][0]	0															
M[i+4][0]	1															
M[i+16][0]	4															
M[i+16][0]	4															

# Examen 2010/11-Q1 (adaptat)

- Quants elements conté una pàgina?
  - $4\text{KB} = 2^{12} \text{ bytes} = 2^{12} / 4 \text{ bytes per element} = 2^{10} \text{ elements} = 4 \text{ files}$
  - A la fila 0 comença la pàgina 0
  - A la fila 4 comença la pàgina 1
  - A la fila 16 comença la pàgina 4
- Cal avançar 4 files per passar a la pàgina següent

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
M[i][0]	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
M[i+4][0]	1	1	1	1	2	2	2	2	3	3	3	3	4	4	4	4
M[i+16][0]	4	4	4	4	5	5	5	5	6	6	6	6	7	7	7	7
M[i+16][0]	4	4	4	4	5	5	5	5	6	6	6	6	7	7	7	7

# Examen 2010/11-Q1 (adaptat)

b) Calcula el nombre d'encerts i fallades del TLB, en tot el bucle, per a cada instrucció:

- Fem un seguiment de les VPN del TLB en ordre LRU

i	0			4			8			12			...
	A[i][0]	A[i+4][0]	A[i+16][0]	A[i][0]	A[i+4][0]	A[i+16][0]	A[i][0]	A[i+4][0]	A[i+16][0]	A[i][0]	A[i+4][0]	A[i+16][0]	
VPN	0	1	4	1	2	5	2	3	6	3	4	7	...
LRU	0	0	0	0	0	4	4	1	5	5	2	6	...
		1	1	4	4	1	1	5	2	2	6	3	...
			4	1	1	2	5	2	3	6	3	4	...
					2	5	2	3	6	3	4	7	...

- Observem que

◦ A[i][0] sols falla a la 1a fila

→ 1 fallada

# Examen 2010/11-Q1 (adaptat)

b) Calcula el nombre d'encerts i fallades del TLB, en tot el bucle, per a cada instrucció:

- Fem un seguiment de les VPN del TLB en ordre LRU

i	0			4			8			12			...
	A[i][0]	A[i+4][0]	A[i+16][0]	A[i][0]	A[i+4][0]	A[i+16][0]	A[i][0]	A[i+4][0]	A[i+16][0]	A[i][0]	A[i+4][0]	A[i+16][0]	
VPN	0	1	4	1	2	5	2	3	6	3	4	7	...
LRU	0	0	0	0	0	4	4	1	5	5	2	6	...
		1	1	4	4	1	1	5	2	2	6	3	...
			4	1	1	2	5	2	3	6	3	4	...
					2	5	2	3	6	3	4	7	...

- Observem que
  - $A[i][0]$  sols falla a la 1a fila  $\rightarrow$  1 fallada
  - $A[i+4][0]$  falla 1 cop cada 4 files  $\rightarrow$  32 fallades

# Examen 2010/11-Q1 (adaptat)

b) Calcula el nombre d'encerts i fallades del TLB, en tot el bucle, per a cada instrucció:

- Fem un seguiment de les VPN del TLB en ordre LRU

i	0			4			8			12			...
	A[i][0]	A[i+4][0]	A[i+16][0]	A[i][0]	A[i+4][0]	A[i+16][0]	A[i][0]	A[i+4][0]	A[i+16][0]	A[i][0]	A[i+4][0]	A[i+16][0]	
VPN	0	1	4	1	2	5	2	3	6	3	4	7	...
LRU	0	0	0	0	0	4	4	1	5	5	2	6	...
		1	1	4	4	1	1	5	2	2	6	3	...
			4	1	1	2	5	2	3	6	3	4	...
					2	5	2	3	6	3	4	7	...

- Observem que
  - $A[i][0]$  sols falla a la 1a fila → 1 fallada
  - $A[i+4][0]$  falla 1 cop cada 4 files → 32 fallades
  - $A[i+16][0]$  falla 1 cop cada 4 files → 32 fallades

# Examen 2010/11-Q1 (adaptat)

c) Suposant que tinguéssim un TLB amb un nombre infinit d'entrades, ¿Quin seria el nombre total de fallades del TLB en aquest bucle?

- Al llarg de les iteracions  $i=0..127$ 
  - $A[i][0]$  visita les 32 pàgines  $0..31$
  - $A[i+4][0]$  visita les 32 pàgines  $1..32$
  - $A[i+16][0]$  visita les 32 pàgines  $4..35$
- Entre els tres accessos, es visiten totes les pàgines  $0..35$
- En total: **36 fallades**