MEMÒRIA CACHE

Introducció

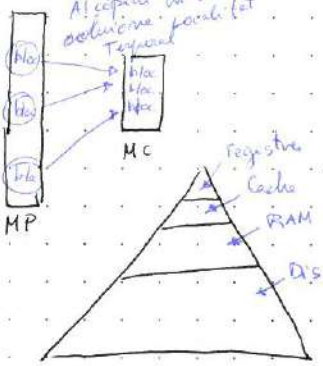
SRAM = "Static". Es fan servir 6 transistors (en DRAM 1) i així fa que sigui més car.
Es fa servir com memòria cache. Més ràpid que DRAM.

Localitat Temporal = Es refereix que si una dada ha estat accedida una vegada, és més probable que es torni a accedir en un futur proper.

Memòria Cache: Situada entre CPU i RAM. Emmagatzema còpies temporals de la RAM. La seva funció és emmagatzemar dades que es faran servir freqüentment.

bloc (subconjunts)

(dades properes)



CPU (Registres) | Cache (MC) (SRAM) | Memòria Principal (MP) (DRAM) | Memòria Secundària (Disc) (SSD/HDD)

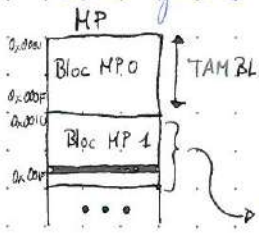
Terminologia

- **Fallada**: Primer la CPU llegeix la cache. Si no està és diu "fallada" (miss). Després es copia de MP→MC el bloc de la dada.
- **Reemplaçament**: Si el bloc que hem de copiar està ocupat amb un bloc antic, es reemplaça.
- **Encert**: La dada sí que està en la MC.

Dissenys bàsics d'una cache

Organització de la memòria en blocs

Cache guarda blocs, no dades "soltes". Bloc té mida TAMBLOC bytes que són potències de 2. (16; 128 són típics)



$$nº\ bloc = \frac{@}{TAMBLOC}$$

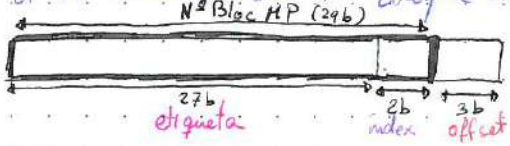
Com que TAMBLOC és pot. de 2, la divisió correspon a desplaçar "eliminem" $\log_2(TAMBLOC)$ bits de menys per dret nº bloc"

$$bloc\ offset = @ \bmod TAMBLOC$$

Això correspon a agafar els $\log_2(TAMBLOC)$ bits menys significatius de la @.

Si es sol·licita [redacted], es còpia tot

Per guanyar velocitat (tot i que no sigui eficient) sempre hi ha relació que si $@\%T = X$ es va sobreescrivint X. Exemple $0x10010000 \rightarrow X=0$ i $0x10010010 \rightarrow X=0$ en cas que $T=16$



etiqueta = Identifica forma única bloc de mem.
índex = Det. quina línia de cache es farà servir per guardar bloc
offset = Identifica part específica dins el bloc

Procediment

1) S'extreu índex de la direcció de memòria
2) Es fa servir índex per accedir línia en cache.
3) Comparar etiq. de la direcció mem amb l'etiq. guardada en aquella línia de la cache.
4) if(etiq==) Hit ; else Miss, Carregar nou bloc sobreescrivint

Exemple: $0x00000028 = (0000\ 0000\ 0010\ 1000)_2$ en Bloc 2^3 [#Bloc MP = 5] i 2^2 línies [#Línia = 1]

00101|000 eliminem els 3 últims i el nº que queda $(0000\ 0101)_2 = (5)_{10}$ i aquest és el nº bloc.

00|01|000 al fer mòdul 4 agafem 2 últims (abans fem div) i aquesta és la línia en Caché

0000 00|01|000 aquesta és l'etiqueta.

Exemple memòria caché

V	Etiqueta	Word 1	Word 2
1	0x0	0x0000 0000	0x0000 0000

4B

#La caché en aquest exemple és de 8B perquè té Word 1 i Word 0.

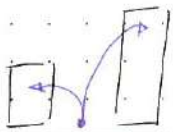
* Si la grandària del bloc de caché és massa gran hi haurà molts miss (Donat que no hi haurà molts blocs al ser SRAM car) i cada miss s'haurà de fer copy gran i això farà perdre temps.

#64Kb és el més òptim (en 2012)

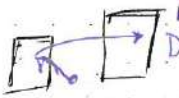
Gestió de les escriptures

Escriptura immediat (Write-Through) [Encert]

S'escriu simultàniament a la memòria caché i a la principal.



Escriptura retardada (Write-back o Copy-back) [Encert]



S'escriu només a la caché. S'escriu a la MP quan la línia de la caché serà sobreescrita.

Escriptura amb assignació (Write allocate) [Miss]

Si es produeix miss, copiem el bloc a la MC (igual que lectura).

Escriptura sense assignació (No-Write allocate) [Miss]

Si es produeix miss, modifiquem contingut en MP, com si no hi hagués caché.

• Escriptura immediata amb assignació (MARS)

[Hit]: Escriu a MP i MC a la vegada.

[Miss]: Còpia bloc MP → MC i després escriu MP i MC a la vegada.

• Escriptura immediata sense assignació

[Hit]: Escriu a MP i MC a la vegada.

[Miss]: Només escriu en bloc de MP (Hi haurà diff contingut MP → Nou, MC → Vell).

#Recorda que SEMPRE MP → MC en cas de miss en lectura

• Escriptura retardada amb assignació

[Hit]: Només s'escriu en MC i BitDirty = 1 (Hi haurà dif cont. MP → Vell, MC → Nou).

[Miss]: Store → Còpia el bloc MP → MC i modifica MC i D = 1.

Load → Còpia MC → MP (si D=1) i després el bloc de MP → MC. Com que és Ld D=0.

Mesura de rendiment

$$t_{acces} = t_h + t_p$$

t_{acces} = Temps de servei d'una referència a memòria

t_h = Temps per determinar si és encert o fallada

t_p = Temps penalització per resoldre referència

Buffer d'Escriptura: Quan tinguem escriptura immediata. Suposarem que no caldrà esperar a MP.

T_p	Immediata sense assig	Ret. amb assig
Lectura - Encert	0	0
Lectura - Fallada	$t_{bloc} + t_h$	bloc mod: $2t_{block} + t_h$ (*) bloc no mod: $t_{bloc} + t_h$
Escriptura - Encert	0	0
Escriptura - Fallada	0	bloc mod: $2t_{block} + t_h$ bloc no mod: $t_{block} + t_h$

(*) 2 perquè hem de portar el dirty a MP.

$t_p = (1-P_e)(t_{block} + t_h)$ ← proporció escriptures

$t_p = P_m(2^* t_{block} + t_h) + (1-P_m)(t_{block} + t_h)$ ← proporció modificats

← no té res a veure / no hi ha diferència entre escriptura i lectura

$$t_{ma} = t_h + m^* t_p$$

(m ← taxa miss)

$$mr_{ins} = \frac{mr}{I}$$

nombre referències per instrucció

$$t_{ma} = \frac{t_{ma:fetch} + (mr_{ins} - 1)^* t_{ma:dades}}{mr_{ins}}$$

$$t_{exe} = M_{ins} {}^* CPI {}^* t_c = (M_{ins} {}^* CPI_{ideal} + M_{fallades} \cdot t_{pm}) t_c$$

(en aquest cas és mitjà; t_c = temps cicle)

Multinivell

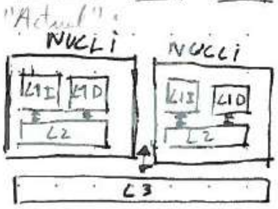
V1:

[CPU] ↔ [L1] ↔ [L2] ↔ [MP]

L1: Blocs petits, molt ràpida. Temps...

L2: Blocs més grans, menys ràpida (Tallafoc per no anar MP). Costosa

"Actual":



(6.1). $HP = 64B$ (2^6) $MC = 16B$ (2^4) $1Bloc = 4B$ (2^2) | $MC = 2^{(2)}$ línies * 2^2 byte/bloc

etiq (2)	#línia (2)	offset (2)

@ 0x39 = 0011|10|01

$OFFSET = 01_2 = 1_{10}$
$\#Línia = 10_2 = 2_{10}$
$\#Etiq = 1110_2 = 14_{10}$

#Bloc MP	binari	# línia MC	hit/miss	
0	0000	0	m	# pq està buit
1	0001	1	m	# pq està buit
0	0000	0	H	
3	0011	3	m	# pq està buit
12	1100	0'	m	# el cont no correspon
10	1010	2	m	# pq està buit
3	0011	3	H	
7	0111	3'	m	# pq cont. no correspon
12	1100	0'	H	
15	1111	3''	m	# pq cont. no correspon.

(6.2). 16b @. Mida total (MC) $= 256B$ (2^8), Mida Bloc $= 16B$ ($2^{(4)}$ offset), Escriptura immediata sense assig.

a) Omple la Taula

$256 / 16 = 2^{(4)}$ → Nº línies (16).

No fem falta — etiq | #lín | offset

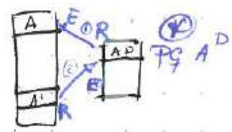
tipus	etiq (Hexa)	índex MC	H/M	#B lleg MP	#B esc MP	Lec dades en MC	Esc dades en MC	
R	45 3 4 (0x45)	0x3	M	16 pq llegeix tot el bloc	0 pq és de lectura	~~NO~~ pq miss SI	Sí pq sempre llegeix	0100.0101|0011|0100
R	45 6 8 (0x45)	0x6	M	16 "	0 "	~~NO~~ " SI	SI "	0100.0101|0110|1000
W	13 A 4 (0x13)	0xA	M	0 pq estem escrivint	~~16~~ 2 pq en vegada [illegible] mod.	NO "	NO gràcies al mode	
W	13 A 8	0xA	M	0	2 pq llegeix	NO pq [illegible]	NO	# Tenc eç R → W
R	35 6 0	0x6'	M	16 pq no està en MC	0 pq llegeix	Sí pq miss	Sí sobreescriu	
W	45 3 C	0x3	H	0 pq és wod	2	NO pq és wod	Sí	immediat
W	60 A 0	0xA'	F	0 pq read	2 pq es write	NO	NO pq és sense assig	només mod MP
R	45 3 C	0x3	E	0 pq està en MC	0 és read	Sí pq és hit	NO pq és read	
W	39 0 0	0x0	F	0	2 pq es write	NO	NO	no porto el mod a MC
R	A2 3 8	0x3'	F	16	0	Sí	SI ← Carrego bloc a MC	Dono resultat desde MC.

pq word són 2B al ser @ de 16b.

pq el ld és read amb el que hi ha a MC

mateix índex però dif etiq.

b) Omple amb | Esc. Ret. amb assig



R	45 3 4	0x3	F	16	0	SI	SI	
R	45 6 8	0x6	F	16	0	SI	SI	
W	13 A 4	0xA^D	F	16	0 pq mod només MC	NO	SI	amb assig (està mod en MC)
W	13 A 8	0xA^D	E	0 pq és W	0 pq està en MC	NO	Si	Retardada (mod de mod)
R	35 6 0	0x6'	F	16	0	Si	SI	
W	45 3 C	0x3^D	E	0 pq és W	0 pq està en MC	NO	Si mod MC	Retardada
W	60 A 0	0xA'(D)	F	16 pq hem de portar MP→MC	16 portar vell bloc MC→MP	Si	SI	Assig + Remplaç
R	45 3 C	0x3^D	E	0	0	Si	NO	
W	39 0 0	0x0	F	16 portar MP→MC	0 mod només MC	NO	Si	Amb assig
R	A2 3 8	0x3'	F	16	16	Si	Si	igual

F. Fixet que el read és de 3^D així que haurem de portar el vell a MP abans de portar MP→MC i després retorn del MC.

(6.7) CPI = 1'5 cicles/ins // t_c = 10ns // nr = 1'6 // dades esc. ret. 1ns // amb anx // Taula.

a) Quin serà t_{ma} (Temps accés mem. mitjà) en cicles?

$$t_{ma} = t_h + m * t_p$$

$$t_{ma} = \frac{t_{ma:f} + (nr_{ins} - 1) * t_{ma:dada}}{nr_{ins}}$$

$$t_{ma:f} = t_{h\,f} + m_f * t_{pf}$$

$$t_{ma:d} = t_{h_d} + m_d * (p_m (2 * t_{p_{mod\,blocs}} + t_h) + (1 - p_m)(t_{p_{nomod}} + t_h))$$

Part block + t_h = t_{p_mod} pero aixo no surt.

t_{p_mod} t_{p_nomod}

$t_{ma:f} = 1 + 0'04 * 10 = 1 + 0'4 = 1'4$

Tq no està en sol?

$t_{ma:d} = 1 + 0'1 * (0'2 (2 * 20 + 1) + (1 - 0'2)(15 + 1)) = 1 + 0'1 (0'2 * 41 + 0'8 * 16) = 1 + 0'1 (8'2 + 12'8) =$

$= 1 + 0'1 (21) = 1 + 2'1 = 3'1$

$$t_{ma} = \frac{1'4 + (1'6 - 1) * 3'1}{1'6} = \frac{1'4 + (0'6) * 3'1}{1'6} = \frac{1'4 + 1'86}{1'6} = \frac{3'26}{1'6} = 2'03 \text{ cicles}$$

b) I t_{exec} en ns?

~~$t_{exec} = (nr_{ins} * CPI_{ideal} + m_f * t_{pm}) * t_c$~~

$$t_{exec:ins} = CPI * t_c$$

$$CPI = CPI_{ideal} + penal_f + penal_d$$

$$penal_f = m_f * t_{pf}$$

$$penal_d = (nr - 1) * m_d (p_m * t_{dm} + (1 - p_m) t_{dmm})$$

$penal_f = 0'04 * 10 = 0'4$

$penal_d = (1'6 - 1) * 0'1 (0'2 * 20 + (0'8 * 15)) = 10'6) * 0'1 * (16) = 0'06 * 16 = 0'86$

$CPI = 1'5 + 0'96 + 0'4 = 2'86$

$t_{exec} = 2'86 * 10 \times 10^{-9} = \boxed{2'86 \times 10^{-9}\ s}$

Pau Bru Ribes

EC Examen de Problemes

Exercici 1 (Examen Final juny 2011)

Considera un computador amb un processador que té amplada de dades i d'adreces de 64 bits, i una memòria cache de dades amb les següents característiques:

- 512 blocs, amb 2 paraules per bloc (paraules de 64 bits)
- correspondència directa
- escriptura immediata sense assignació

a) Quina és la capacitat en bytes per a dades de la memòria cache? 8192 Bytes

b) Indica el rang de bits de l'adreça que especifiquen l'índex a la memòria cache [4, 12]

c) Indica el rang de bits de l'adreça que especifiquen l'etiqueta [13, 63]

d) Quants bits d'emmagatzematge per a etiquetes i bits de control fan falta en total per cada entrada de la memòria cache? 52 b

e) Considera el següent programa en alt nivell, que s'executa en aquest computador:

```
int V[6];            /* un int ocupa 64 bits */
main() {
    int i, tmp;        /* variables ubicades en registres */

    tmp = V[0];
    for (i=1; i<6; i++) V[i-1] = V[i];
    V[5] = tmp;
}
```

@	V/w	H/M
0	r	m
4	r	m
0	w	h
8	r	m
4	w	h
12	r	m
8	w	h
16	r	m
12	w	h
20	r	m
16	w	h
20	w	h

Tenint en compte que el vector V està emmagatzemat a partir de l'adreça 0, indica la seqüència d'adreces (en hexadecimal) dels accessos a memòria de dades que genera l'execució del programa, especificant per cada una: si és lectura o escriptura (R/W) i si produeix un encert o fallada (hit/miss) a la cache.

Exercici 2 (Examen Final gener 2013)

Considera el següent programa:

```
int M[4][2];            /* adreça base d'M = 0 */
int V[2];
main() {
    int i,j;            /* emmagatzemades en registres */
    for (i=0; i<4; i++)
        for (j=0, j<2; j++)
            M[i][j] = M[i][j]+V[j];
}
```

que s'executa en un computador MIPS que disposa d'una memòria cache de dades, inicialment buida, de correspondència directa i política d'escriptura retardada amb assignació, que conté 4 blocs i on els blocs són de 8 bytes.

Emplena la següent taula, que mostra la seqüència de les 12 primeres referències a memòria (E: escriptura/ L: lectura) corresponent al programa.

(*1) Tanta de MP→MC a mode MC.
(*2) Com que és V[1] còpia V[0] i V[1] i hi havia D=1
(*3) Només mode MC.

(*) Fixe't que un bloc són 8B. Quan fa L1 M[0][0] al fer 4B, còpia també M[0][1] a la MC.

□ No entra pq és línia 0 i no 1 (Hauria de copiar V[1]; V[2] ~~tot~~ que no existeix no?
~~□ No entra pq 8B de les 2~~

	element accedit	línia de MC	hit/miss	bytes llegits d'MP	bytes escrits a MP
L	M[0][0]	0	miss	8	0
L	V[0]	32%4=0	miss	8	0
E	M[0][0]	0	miss	8	0
L	M[0][1]	0	hit	0	0
L	V[1]	1	miss	8	8 (*2)
E	M[0][1]	0	miss	8	0
L	M[1][0]	1	miss	8	0
L	V[0]	0	miss	8	8
E	M[1][0]	1	hit	0	0 (*3)
L	M[1][1]	1	hit	0	0
L	V[1]	0	hit	0	0
E	M[1][1]	1	hit	0	0

Table 1: TLB

Exercici 3 (Examen Final juny 2012)

Considera el següent programa:

```
int M[F][C];              /* adreça base d'M = 0 */

main() {
int i,j;                  /* emmagatzemades en registres */

    for (i=0; i<F-4; i++)
        for (j=0, j<C; j++)
            M[i][j] = M[i+4][j];
}
```

que s'executa en un computador MIPS que disposa d'una memòria cache de dades, inicialment buida, de correspondència directa, que conté 16 blocs i on els blocs són de 16 bytes.

Omple una taula especificant el nombre de referències, el nombre de fallades i el nombre de bytes transferits a/des de MP per aquests dos casos:

- MC amb política d'escriptura immediata sense assignació; matriu M amb F=16 i C=16

M[0][0] → M[0][1]
M[0][4] → M[0][7]
M[0][8] → M[0][11]
M[0][12] → M[0][15]

M[0][0] → M[4][0]
M[0][4] → M[4][4]

Podem veure que cap fet a la cache

- MC amb política d'escriptura retardada amb assignació; matriu M amb F=8 i C=8

Esc. imm. sense. assig F=16 i C=16

	r/w	b/m	#r	#w
* M[4][0]	R	m	16	0
M[0][0]	W	m	0	4
M[4][1]	R	h	0	0
M[0][1]	W	h	0	4
5 m 3 h M[4][2]	R	h	0	0
M[0][2]	W	m	0	4
M[4][3]	R	h	0	0
M[0][3]	W	m	0	4
* M[4][4]	R	m	16	0
M[0][4]	W	m	0	16
M[4][5]	R	h	0	0
M[0][5]	W	m	0	16
M[4][6]	R	h	0	0
M[0][6]	W	m	0	16

16 B * (12 * 16) = 30 + 2 = 6144

Nº ins = 192 = 12 * 16

Nº ref = 2 * 192 = 384 = Nº ref

0 < 16 < 44 = 12 → 12 línies

0 < 16 però 4 repes per fila

12 * 4 * (5) = 240 miss

12 * 4 * (16 + 4 + 4 + 4 + 4)

12 * 4 * (32) = 1536 B

Esc retardat amb assig. F=8 i C=8

	b/m	#r	#w
R M[4][0]	m	16	0
W M[0][0]	m	16	0
R M[4][1]	h	0	0
W M[0][1]	h	0	0
R M[4][2]	h	0	0
W M[0][2]	h	0	0
R M[4][3]	h	0	0
W M[0][3]	h	0	0
R M[4][4]	m	16	0
W M[0][4]	m	16	0
R M[4][5]	h	0	0
W M[0][5]	h	0	0
R M[4][6]	h	0	0
W M[0][6]	h	0	0

Nº ref = (8 * 8) * 2 = 64 * 2 = 128 = Nº ref

Hi haurà 16 miss pq per cada bloc hi ha miss i no hi ha coherència

16 * 16 = 256 B

Ens fixem que tot i que hi hagi Dirty a 1 en alguns blocs mai hi ha sobreescriu.

Pau Bru Ribes

EC Examen de Problemes [redacted]

Exercici 1 (Examen Final 2011-2012 Q1)

Suposem que tenim un processador de 32 bits amb una memòria cache de dades de 8 KB associativa per conjunts de 2 vies, on cada bloc té 16 bytes, i que es segueix l'algorisme de reemplaçament LRU.

- Calcula el nombre de fallades de la cache en executar els següent programa, suposant que la cache té la política d'escriptura immediata sense assignació, i que la memòria cache és inicialment buida. L'adreça base del vector A és 0.

```
int A[1024];
int B[1024];
int C[1024];

void main() {
  int i;
   for (i=0;i<1024;i++)
      A[i]=B[i]+C[i];
}
```

fallades A= 1024
fallades B= 256
fallades C= 256

- Fes el mateix per al següent programa:

```
int A[512];
int B[512];
int C[512];

void main() {
int i;
    for (i=0;i<512;i++)
        A[i]=B[i]+C[i];
}
```

fallades A= 512
fallades B= 128
fallades C= 128

- Repeteix els dos apartats anteriors considerant ara que la cache té una política d'escriptura retardada amb assignació.

fallades A= 1024
fallades B= 1024
fallades C= 1024

fallades A= 128
fallades B= 128
fallades C= 128

Exercici 2 (Examen Final 2013-2014 Q1)

Suposem que tenim un processador de 32 bits amb una memòria cache de dades de 256 bytes, on cada bloc té 16 bytes. Suposem que executem els següents programes.

```
//programa A
int M[4][64];

void main() {
int i, j; //en registres
  for (i=0;i<4;i++)
    for (j=0;j<64;j++)
      M[i][j]= 0;
}
```

```
//programa B
int M[4][64];

void main() {
int i, j; //en registres
  for (j=0;j<64;j++)
    for (i=0;i<4;i++)
      M[i][j]= M[i][j]+1;
}
```

Calcula el nombre de fallades de la cache suposant que la memòria cache és inicialment buida. L'adreça base de la matriu M és 0.

- Suposant que la cache és de correspondència directa i té la política d'escriptura retardada amb assignació.

  falladesA= 64 falladesB= 256 (4*64)

- Suposant que la cache és associativa per conjunts de 4 vies (algorisme de reemplaçament LRU), i que té la política d'escriptura immediata sense assignació.

  falladesA= 256 falladesB= 64