

Exe 2 : Cache D.P./S.A en Write-Back

On considère un cache de 1024 octets avec une taille de bloc (ou ligne) de 16 octets.

la mémoire est adressée par octets et les adresses sont sur 32 bits.

le cache est géré en écriture différée (un défaut en écriture entraîne un changement du bloc correspondant), en réécriture (Write-Back, une donnée n'est modifiée en mémoire principale que lorsque la ligne de cache correspondante est évacuée du cache) et avec une politique de remplacement LRU (Least Recently Used).

Q1/ Structure des adresses :

a/ Donner la structure de l'adresse physique vue par le cache en précisant le nombre de bits de chaque champ dans le cas d'un cache Direct Mapped, d'un cache Set-Associative de degré 2, pareil pour un degré 4 :

Taille.Bloc = 16 octets et Taille.cache = 1024 octets = 64 lignes de 16 octets.

① Dans le cas d'un cache en Direct-Mapping :

$$\text{Nb.bits(Dep)} = \log_2(\text{Taille.Bloc}) = \log_2(16) = \log_2(2^4) = 4 \text{ bits.}$$

$$\text{Nb.bits(\#Ens)} = \log_2\left(\frac{\text{Taille.cache}}{\text{Taille.Bloc} \times \text{Deg.A}}\right) = \log_2\left(\frac{2^{10}}{2^4 \times 2^0}\right) = 6 \text{ bits.}$$

$$\text{Nb.bits(Tag)} = 32 - \text{Nb.bits(Dep)} - \text{Nb.bits(\#Ens)} = 32 - 10 = 22 \text{ bits.}$$

② Dans le cas d'un cache en Set-Associatif de degré 2 :

$$\text{Nb.bits(Dep)} = 4 \text{ bits ; Nb.bits(\#Ens)} = \log_2\left(\frac{2^{10}}{2^4 \times 2^1}\right) = 5 \text{ bits ; Nb.bits(Tag)} = 23 \text{ bits.}$$

③ Dans le cas d'un cache en Set-Associatif de degré 4 :

$$\text{Nb.bits(Dep)} = 4 \text{ bits ; Nb.bits(\#Ens)} = 4 \text{ bits ; Nb.bits(Tag)} = 24 \text{ bits.}$$

b/ Donner un schéma du cache mettant en évidence l'utilisation de chaque champs dans le cas d'un cache associatif par ensemble de 2 blocs (degré 2).

Quelle est l'utilité des deux niveaux d'associativité ?

Pour le schéma, il suffit de se référer à celui de l'Exe1, Q1, b en dédoublant le Répertoire et la section Data.

L'utilité des deux niveaux d'associativité est de réduire les Miss de conflits en exploitant la localité temporelle.

Q2/ Analyse d'une application : Soient X, Y et Z des tableaux de réels en double-précision (8 octets), X est implanté à partir de l'adresse $0x10010$, Y à partir de $0x20210$ et Z à partir de $0x020410$. On étudie la boucle :

for ($i=0$; $i < N$; $i++$)
 $\quad Z[i] = X[i] + Y[i];$

En supposant que $N=3$, indiquer pour chaque accès mémoire (lecture & écriture), si c'est un succès ou un échec (on traite les accès dans l'ordre X_i, Y_i, Z_i) .

c/ Dans le cas d'un cache en Direct-Mapping en donnant pour chaque référence le numéro de la case de cache où elle est stockée.

Structure-Adresse : 22, 6, 4

⊕ Pour X : $\begin{cases} X_0 & X_1 & X_2 \\ 0x10010 & 0x10018 & 0x10020 \end{cases}$

⊕ Pour Y : $\begin{cases} Y_0 & Y_1 & Y_2 \\ 0x20210 & 0x20218 & 0x20220 \end{cases}$

⊕ Pour Z : $\begin{cases} Z_0 & Z_1 & Z_2 \\ 0x020410 & 0x020418 & 0x020420 \end{cases}$

Direct-Mapping

Case 1 (80)		Case 0 (80)		
X_1	X_0			0
				1
				2
		...		
Y_1	Y_0			33
		...		
				63

Effets des instructions et détail de (#Eus, Dep):

Load $X_0 \rightarrow (0b000001, 0b0000) \Rightarrow$ Miss comp. clean .

Load $Y_0 \rightarrow (0b100001, 0b0000) \Rightarrow$ Miss comp. clean .

Store $Z_0 \rightarrow (0b000001, 0b0000) \Rightarrow$ Miss comp. clean .

Load $X_1 \rightarrow (0b000001, 0b1000) \Rightarrow$ Miss comp. dirty .

Load $Y_1 \rightarrow (0b100001, 0b1000) \Rightarrow$ Hit .

Store $Z_1 \rightarrow (0b000001, 0b1000) \Rightarrow$ Miss comp. clean .

Load $X_2 \rightarrow (0b000010, 0b0000) \Rightarrow$ Miss comp. clean .

Load $Y_2 \rightarrow (0b100010, 0b0000) \Rightarrow$ Miss comp. clean .

Store $Z_2 \rightarrow (0b000010, 0b0000) \Rightarrow$ Miss comp. clean .

On a 9 accès : 1 Hit et 8 Miss (dont 7 clean et 1 dirty) .

Z_1	Z_0			0
				1
				2
		...		
Y_1	Y_0			33
		...		
				63

X_1	X_0			0
?	X_2			1
				2
		...		
Y_1	Y_0			33
?	Y_2			34
				63

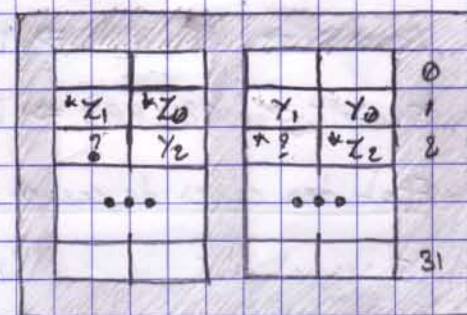
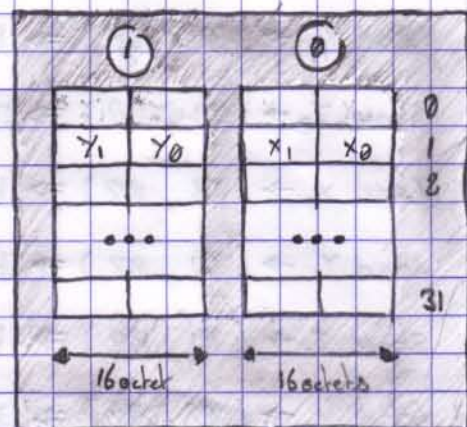
d/ Dans le cas d'un cache en Set-Associatif de degré 2 en donnant pour chaque référence le numéro d'ensemble où elle est stockée.

Structure Adresse : 23, 5, 4

2-Way Associative

Effet des instructions et détail de (#Ens, rep) :

Load $X_0 \rightarrow (0b\ 0\ 0001, 0b\ 0000) \Rightarrow$ Miss comp clean.
 Load $Y_0 \rightarrow (0b\ 0\ 0001, 0b\ 0000) \Rightarrow$ Miss comp clean.
 Store $Z_0 \rightarrow (0b\ 0\ 0001, 0b\ 0000) \Rightarrow$ Miss comp clean.
 Load $X_1 \rightarrow (0b\ 0\ 0001, 0b\ 1000) \Rightarrow$ Miss comp clean.
 Load $Y_1 \rightarrow (0b\ 0\ 0001, 0b\ 1000) \Rightarrow$ Miss comp dirty.
 Store $Z_1 \rightarrow (0b\ 0\ 0001, 0b\ 1000) \Rightarrow$ Miss comp clean.
 Load $X_2 \rightarrow (0b\ 0\ 0010, 0b\ 0000) \Rightarrow$ Miss comp clean.
 Load $Y_2 \rightarrow (0b\ 0\ 0010, 0b\ 0000) \Rightarrow$ Miss comp clean.
 Store $Z_2 \rightarrow (0b\ 0\ 0010, 0b\ 0000) \Rightarrow$ Miss comp clean.



État des cases de cache de la famille 1 :

Instruction i	init	LX_0	LY_0	SZ_0	LX_1	LY_1	SZ_1	LX_2	LY_2	SZ_2
Case 00 après i	\emptyset	X_0	X_0	$*Z_0$	$*Z_0$	Y_0	Y_0	Y_0	Y_0	Y_0
Case 01 après i	\emptyset	X_1	X_1	$*Z_1$	$*Z_1$	Y_1	Y_1	Y_1	Y_1	Y_1
Case 10 après i	\emptyset	\emptyset	Y_0	Y_0	X_0	X_0	$*Z_0$	$*Z_0$	$*Z_0$	$*Z_0$
Case 11 après i	\emptyset	\emptyset	Y_1	Y_1	X_1	X_1	$*Z_1$	$*Z_1$	$*Z_1$	$*Z_1$

État des cases de cache de la famille 2 :

Instruction i	init	LX_0	LY_0	SZ_0	LX_1	LY_1	SZ_1	LX_2	LY_2	SZ_2
Case 00 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	X_2	X_2	$*Z_2$
Case 01 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	$?$	$?$	$*?$
Case 10 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	Y_0	Y_0
Case 11 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	$?$	$?$

On a 9 accès : 9 Miss dont 8 clean et 1 dirty.

e/ Dans le cas d'un cache associatif par ensemble de 4 blocs et donner pour chaque référence le numéro de l'ensemble où elle est stockée.

Structure Adresse : 24, 4, 4

4-Way Associative

Effet instruction et détail de (#Ens, Dep):

load $X_0 \rightarrow (01, 00) \Rightarrow$ Miss comp clean.

load $Y_0 \rightarrow (01, 00) \Rightarrow$ Miss comp clean.

store $X_0 \rightarrow (01, 00) \Rightarrow$ Miss comp clean.

load $X_1 \rightarrow (01, 08) \Rightarrow$ Hit.

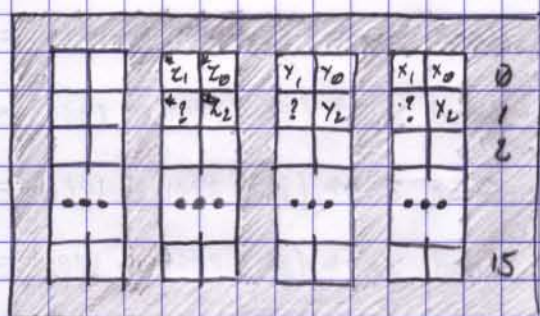
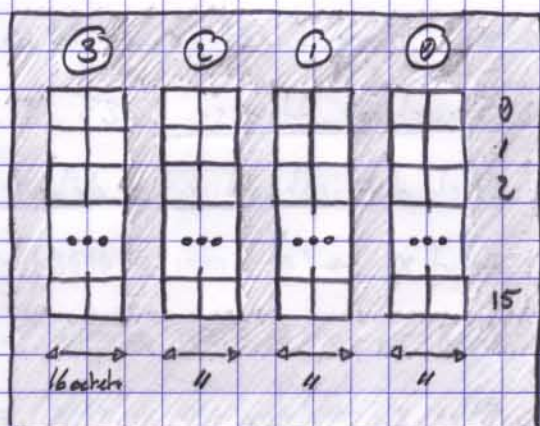
load $Y_1 \rightarrow (01, 08) \Rightarrow$ Hit.

store $Z_1 \rightarrow (01, 08) \Rightarrow$ Hit.

load $X_2 \rightarrow (02, 00) \Rightarrow$ Miss comp clean.

load $Y_2 \rightarrow (02, 00) \Rightarrow$ Miss comp clean.

store $Z_2 \rightarrow (02, 00) \Rightarrow$ Miss comp clean.



Etat des cases de caches des familles 1 et 2:

Instruction i	init	LX_0	LY_0	SX_0	LX_1	LY_1	SX_1	LX_2	LY_2	SX_2
Case 00 après i	\emptyset	X_0	"	"	"	"	"	"	"	"
Case 01 après i	\emptyset	X_1	"	"	"	"	"	"	"	"
Case 10 après i	\emptyset	\emptyset	Y_0	"	"	"	"	"	"	"
Case 11 après i	\emptyset	\emptyset	Y_1	"	"	"	"	"	"	"
Case 20 après i	\emptyset	\emptyset	\emptyset	$*X_0$	"	"	"	"	"	"
Case 21 après i	\emptyset	\emptyset	\emptyset	$*X_1$	"	"	"	"	"	"
Case 30 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
Case 31 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
Case 00 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	X_2	"	"
Case 01 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	?	"	"
Case 10 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	Y_2	"
Case 11 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	?	"
Case 20 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	$*Z_2$
Case 21 après i	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	$*?$

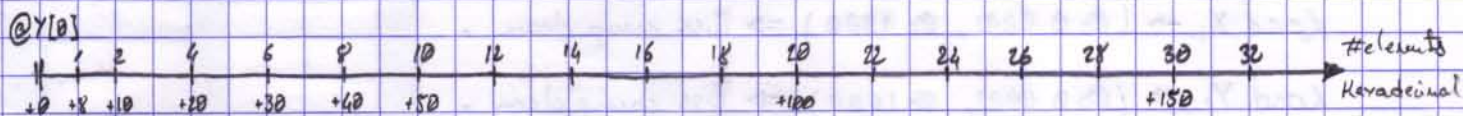
Famille 1

Famille 2

f) Pour une valeur de N quelconque, quel est le taux de Miss pour les 3 configurations de caches considérés ?

Concernant les adresses de nos structures, on remarque que N est limité : on a $@X[0] = 0 \times 10010$, $@Y[0] = 0 \times 20210$ et $@Z[0] = 0 \times 20410$ avec une taille d'élément de 8 octets, comme $@Z[0] - @Y[0] < @Y[0] - @X[0]$, cette limite est définie par $@Z[0] - @Y[0] = 0 \times 200$ octets.

De plus, comme 2 éléments prennent $0 \times 10 = 16$ octets, on a donc le graphe suivant :



Le tableau Y peut donc contenir au plus 39 éléments avant de chevaucher Z .
Donc $N \leq 39$ éléments.

④ Dans le cas d'un cache Direct-Mapped :

$$\tau_{\text{Miss}} = \frac{\# \text{Miss}}{\# \text{Acces}}$$

On change d'ensemble tout les 2 éléments pour nos 3 structures, donc pour une utilisation d'un même ensemble, on a : 2 itérations et 6 accès mémoires dont 1 Hit et 5 Miss.

\Rightarrow Nous n'avons aucune Miss de conflit causée par Y par la borne de N .

\Rightarrow Nous avons un Hit causé par Y dans une itération sur 2.

$\Rightarrow \# \text{Acces} = 3N$; $\# \text{Hit} = \lfloor \frac{N}{2} \rfloor$; $\# \text{Miss} = \# \text{Acces} - \# \text{Hit} = 3N - \lfloor \frac{N}{2} \rfloor$.

\Rightarrow Donc $\tau_{\text{Miss}} = \frac{3N - \lfloor \frac{N}{2} \rfloor}{3N} = 1 - \frac{\lfloor \frac{N}{2} \rfloor}{3N}$.

④ Dans le cas d'un cache 2-Way Associative :

\Rightarrow Peu importe la valeur de N , on a $\tau_{\text{Miss}} = 100\%$.

④ Dans le cas d'un cache 4-Way Associative :

$\# \text{Acces} = 3N$; $\# \text{Hit} = 3 * \lfloor \frac{N}{2} \rfloor$; $\# \text{Miss} = 3N - 3 * \lfloor \frac{N}{2} \rfloor = 3 * (N - \lfloor \frac{N}{2} \rfloor)$.

$\Rightarrow \tau_{\text{Miss}} = \frac{3 * (N - \lfloor \frac{N}{2} \rfloor)}{3N} = \frac{N - \lfloor \frac{N}{2} \rfloor}{N} = 1 - \frac{\lfloor \frac{N}{2} \rfloor}{N}$.

On étudie maintenant la boucle suivante et on considère uniquement le cas d'un cache associatif par ensemble de 2 blocs (2-Way Associative) :

```
for (i = 0 ; i < N ; i++)
    ↓
    Z[i] = X[i+1] + Y[i+1];
```

g/ Pour chaque accès mémoire, indiquer si il s'agit d'un succès ou d'un échec et donnez pour chaque références le numéro de l'ensemble où elle est stockée en supposant $N=4$. Quel est le taux de Miss pour une valeur quelconque de N ?

Structure Adresse : 23, 5, 4

2-Way Associative

Effet des instructions et détail de (#Eus, Dep):

Load $X_1 \rightarrow (0b00001, 0b1000) \Rightarrow$ Miss comp clean.

Load $Y_1 \rightarrow (0b00001, 0b1000) \Rightarrow$ Miss comp clean.

Store $Z_0 \rightarrow (0b00001, 0b0000) \Rightarrow$ Miss comp clean.

Load $X_2 \rightarrow (0b00010, 0b0000) \Rightarrow$ Miss comp clean.

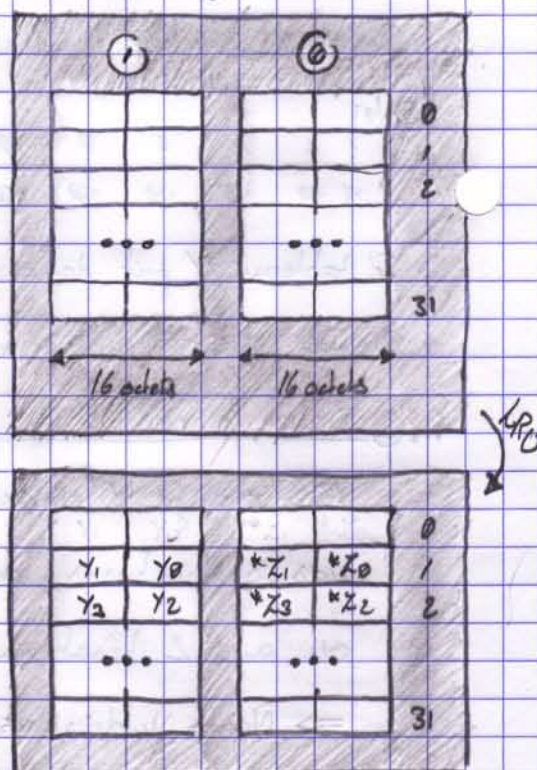
Load $Y_2 \rightarrow (0b00010, 0b0000) \Rightarrow$ Miss comp clean.

Store $X_1 \rightarrow (0b00001, 0b1000) \Rightarrow$ Hit.

Load $X_3 \rightarrow (0b00010, 0b1000) \Rightarrow$ Hit.

Load $Y_3 \rightarrow (0b00010, 0b1000) \Rightarrow$ Hit.

Store $Z_2 \rightarrow (0b00010, 0b0000) \Rightarrow$ Miss comp clean.



Etat des cases de cache des ensembles 1 et 2 :

Instruction :	init	LX_1	LY_1	SX_0	LX_2	LY_2	SX_1	LX_3	LY_3	SX_2
Case 00 apres i	\emptyset	X_0	X_0	$*Z_0$	$*Z_0$	$*X_0$	$*X_0$	$*X_0$	$*Z_0$	$*Z_0$
Case 01	\emptyset	X_1	X_1	$*Z_1$	$*Z_1$	$*X_1$	$*X_1$	$*X_1$	$*Z_1$	$*Z_1$
Case 10	\emptyset	\emptyset	Y_0	Y_0	Y_0	Y_0	Y_0	Y_0	Y_0	Y_0
Case 11	\emptyset	\emptyset	Y_1	Y_1	Y_1	Y_1	Y_1	Y_1	Y_1	Y_1
Case 00	\emptyset	\emptyset	\emptyset	\emptyset	X_2	X_2	X_2	X_2	X_2	$*Z_2$
Case 01	\emptyset	\emptyset	\emptyset	\emptyset	X_3	X_3	X_3	X_3	X_3	$*Z_3$
Case 10	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	Y_2	Y_2	Y_2	Y_2	Y_2
Case 11	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	Y_3	Y_3	Y_3	Y_3	Y_3

End 1

End 2

Exe 3 : Comparaisons des Systèmes avec et sans caches.

On s'intéresse à une machine monoprocesseur construite autour du processeur Itip pipeline à 5 étages étudié en cours. On souhaite étudier l'influence de la mémoire sur la performance globale du système. Le nombre moyen de cycles par instruction (CPI) dans l'hypothèse d'un système mémoire idéale (càd les lectures et écritures prennent 1 cycle) est $CPI_{idéale} = 1,1 \text{ cy/instr.}$

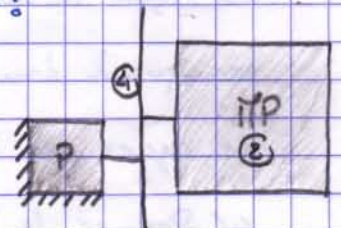
Les instructions Load représentent 80% des instructions exécutées et les instructions Store représentent 10%.

La mémoire physique du système a une capacité de 64 Miga-octets et elle est adressée par octets. Le nombre de cycles nécessaires pour une lecture est de 10 cycles processeur, il est de 4 cycles pour une écriture.

Q1/ Système sans cache et Comparaison :

a/ Par rapport au système idéal, quelle est l'augmentation du nombre de cycles par instruction due aux écritures (noté δW) ? Quelle est l'augmentation du nombre de cycles par instruction due aux lectures (noté δR) ?

Ici, nous n'avons pas de cache ! Le processeur est directement connecté au bus système.



Les lectures prennent 10 cycles : 4 cy pour l'envoi d'une adresse, puis 2 cycles pour lire dans la mémoire principale et enfin 4 cycles pour l'envoi de la donnée de la MP vers le processeur. Les écritures ne prennent que 4 cycles pour l'envoi de l'adresse à la MP.

① Augmentation du CPI à cause des écritures :

$$\Rightarrow \delta W = \frac{\#C}{\#I} = \frac{\#C}{\#Ecrit} \times \frac{\#Ecrit}{\#I} = (\#C_{écrit} - \#C_{idéale}) \times \frac{\#Ecrit}{\#instr} = (4-1) \times \frac{10}{100} = 0,3 \text{ cy/instr.}$$

② Augmentation du CPI à cause des lectures (⚠ chaque instruction est lue ⚠) :

$$\Rightarrow \delta R = \frac{\#C}{\#I} = \frac{\#C}{\#lecture} \times \frac{\#lecture}{\#I} = (\#C_{lect} - \#C_{idéale}) \times \frac{\#lecture}{\#instr} = (10-1) \times \frac{20+100}{100} = 10,8 \text{ cy/instr.}$$

b/ Quel est le nombre moyen de cycles par instruction (CPI)?

$$CPI_{ss.cache} = CPI_{ideal} + \overline{JW} + \overline{JR} = 1,1 + 0,3 + 10,8 = 12,2 \text{ cy/inst.}$$

Pour minimiser la dégradation de performance due à la lenteur de la mémoire, on introduit un cache à deux niveaux d'associativité.

(2-Way Associative) d'une capacité de 16 kilo-accès où les données et les instructions sont mises en commun (cache unifié).

le temps d'accès du processeur au cache est de 1 cycle. La taille d'un bloc est de 16 octets. On utilise une stratégie d'écriture simultanée dans la mémoire et dans le cache (Write Through) sans mise à jour du cache si la donnée écrite n'est pas présente dans le cache (pas de Miss d'écriture).

Il n'y a pas de Buffer d'écriture Postée.

la durée d'une lecture en cas de Miss est de 23 cycles (22 cycles pour le transfert d'une ligne de cache de la MP vers le cache et 1 cycle pour envoyer la donnée du cache au processeur).

Q2/ Système avec cache unifié en WT sans BEP et comparaison :


On suppose que le taux de Miss est de 5%. En cas de Miss, le processeur ne reçoit la donnée demandée qu'après le chargement complet du bloc manquant dans le cache. On cherche à calculer le nombre moyen de cycles par instruction (CPI).

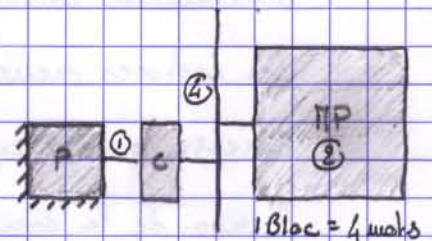
c/ Donner la taille des champs Tag, Index et Offset.

$$\text{Nb.bits(Index)} = \log_2\left(\frac{\text{Taille-cache}}{\text{Taille-Bloc} \times \text{Deg-A}}\right) = \log_2\left(\frac{2^4 \times 2^{10}}{2^4 \times 2^1}\right) = 9 \text{ bits.}$$

$$\text{Nb_bits(Offset)} = \log_2(\text{Taille_Bloc}) = \log_2(2^4) = 4 \text{ bits.}$$

$$\text{Nb_bits}(\text{Tag}) = \text{Nb_bits}(\text{@}) - \text{Nb_bits}(\text{Index}) - \text{Nb_bits}(\text{offset})$$

 $\rightarrow = \log_2(\text{capacite_MP}) - 9 - 4$
 $= \log_2(64 \times 2^{10} \times 2^0) - 9 - 4$
 $= \log_2(2^6 \times 2^{10}) - 13 = 26 - 13 = 13 \text{ bits.}$



④ Caches Séparés :

↓ (+) Simple (+) 2 access/cycles

④ Caches Unifiés :

⊕ Complexes ⊖ Ions/cycle

④ Puro sur Dt ④ Sample & Dyne.

d/ Par rapport au système idéal, quelle est l'augmentation du CPI due aux écritures (noté δW) ? Quelle est l'augmentation du nombre moyen de cycles par instruction (\overline{CPI}) due aux lectures (noté δR) ?

#Cwrite = 4 cycles car il n'y a pas de Miss d'écriture en WT sans BOP.

⚠ $\#C_{read} = \overline{C}_{Hit} \times \#C_{Hit-read} + \overline{C}_{Miss} \times \#C_{Miss-read} = 0,95 \times 1 + 0,05 \times 23 = 2,1 \text{ cycles}.$

$\Rightarrow \delta W = (\#C_{write} - \#C_{ideal}) \times \frac{\#Write}{\#Instr} = (4 - 1) \times \frac{10}{100} = 0,3 \text{ cycles/instr.}$

$\Rightarrow \delta R = (\#C_{read} - \#C_{ideal}) \times \frac{\#Read}{\#Instr} = (2,1 - 1) \times \frac{20+100}{100} = 1,1 \times 1,2 = 1,32 \text{ cycles/instr.}$

e/ Quel est le nombre de cycles par instruction (CPI) ?

$CPI = CPI_{ideal} + \delta W + \delta R = 1,1 + 0,3 + 1,32 = 2,72 \text{ cycles/instructions.}$

Pour limiter la pénalité introduite par les Miss de lecture, on introduit une cache secondaire (cache de niveau 2) entre le cache unifié et la mémoire.

Le cache secondaire a une capacité de 512 Ko. La taille du bloc est de 16 octets.

Il est de type Direct-Mapped et tout les blocs contenus dans le cache primaire sont contenus dans le cache secondaire (caches inclusifs).

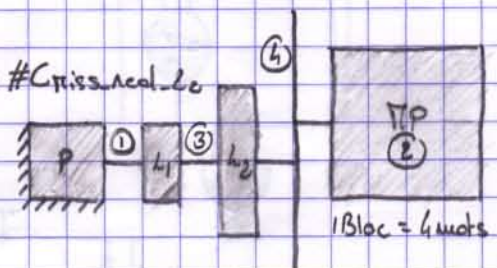
Q3/ Système avec deux niveaux de caches et comparaison :

Le taux de Miss dans le cache secondaire est le pourcentage de requêtes émises par le cache primaire qui font Miss dans le cache secondaire, il est de 15%.

Le temps d'accès au cache secondaire est de 3 cycles (idem pour le temps de transfert d'un bloc du cache secondaire dans le cache primaire).

f/ De point de vue du processeur, quelle est la durée d'une lecture en cas de Miss dans le cache primaire ?

$\#C_{Miss-read-L1} = \overline{C}_{Hit-read-L1} \times \#C_{Hit-read-L1} + \overline{C}_{Miss-read-L1} \times \#C_{Miss-read-L1}$
 $= 0,85 \times (3+1) + 0,15 \times (1+3+22)$
 $= 3,4 + 3,9 = 7,3 \text{ cycles.}$



g/ Par rapport au système idéal, quelle est l'augmentation du nombre moyen de cycles par instruction due aux écritures (noté δW) ?

Quelle est l'augmentation du nombre moyen de cycles par instructions due aux lectures (noté δR) ?

#C_{write} = 4 cycles car il n'y a pas de miss d'écriture en WT.

$$\#C_{read} = T_{Hit-read-L_1} \times \#C_{Hit-read-L_1} + T_{Miss-read-L_1} \times \#C_{Miss-read-L_1} \\ = 0,95 \times 1 + 0,05 \times 7,3 = 0,95 + 0,365 = 1,315 \text{ cycles.}$$

$$\Rightarrow \delta W = (\#C_{write} - \#C_{ideal}) \times \frac{\#Write}{\#instr} = (4 - 1) \times \frac{10}{100} = 0,3 \text{ cycles/instr.}$$

$$\Rightarrow \delta R = (\#C_{read} - \#C_{ideal}) \times \frac{\#Read}{\#instr} = (1,315 - 1) \times \frac{20+100}{100} = 0,315 \times 1,2 = 0,378 \text{ cycles/instr.}$$

h/ Quel est le nombre de cycles par instructions (CPI) ?

$$CPI = CPI_{ideal} + \delta W + \delta R = 1,1 + 0,3 + 0,378 = 1,778 \text{ cycles/instruction.}$$

G4/ Système avec deux niveaux de caches munis de B&P et Comparaison :

On cherche enfin à diminuer la pénalité introduite par les écritures.

Quelle solution suggérez-vous ? Quel bénéfice peut-on espérer sur le CPI ?

Si on met un Buffer d'Écriture Postée dans le cache L₁, ou dans les deux caches, l'augmentation du CPI due aux écritures est de 0 cycles tant que le débit d'écriture est inférieur au débit d'évacuation.

Schéma de l'utilisation d'une adresse dans un cache 2-way-Associative

