

TD 4 - Caches - Correction

1 Etiquette et Index de cache

Un processeur a 2 Gio de mémoire principale. Pour les différents caches ci-dessous :

1. cache de 2 Mio à correspondance directe et écriture simultanée avec des blocs de 16 octets
2. cache de 4 Mio à correspondance directe, réécriture et blocs de 32 octets
3. cache de 4 Mio associatif 4 voies (4 blocs par ensemble), réécriture et blocs de 32 octets

1.1 Quelle est la décomposition d'une adresse mémoire (nombre de bits des différentes parties) ?

Rappels :

- l'index indique dans quelle ligne va la donnée.
- l'adresse dans le bloc (ou offset) indique où est la donnée dans la ligne
- l'étiquette sert à comparer si la donnée que l'on cherche est bien celle qui est stockée actuellement dans le cache ou non.

Mémoire de 2 Gio = $2 * 2^{30} = 2^{31}$. Les adresses font donc 31 bits.

1. Cache de 2 Mio = $2 * 2^{20} = 2^{21}$ octets. Comme une ligne = 16 octets, il y a $2^{21}/2^4 = 2^{17}$ lignes. L'index étant la partie qui permet de savoir à quelle ligne est la donnée dans le cache, il faut 17 bits d'index.
La ligne de cache étant de $16 = 2^4$ octets, pour savoir à quel endroit est la donnée dans la ligne, il faut 4 bits d'adresse dans bloc (offset).
Enfin, l'étiquette fait $31 - 17 - 4 = 10$ bits
2. Même explication que précédemment.
17 bits d'index.
5 bits d'adresse dans bloc (offset).
9 bits d'étiquette. ($31 - 17 - 5$)
3. Même explication que précédemment, sauf pour le nombre de lignes.
15 bits d'index. En effet, comme le cache est associatif par 4 voies, $4 = 2^2$ lignes ont la même adresse. Donc les 2^{17} lignes sont par groupes de 2^2 soit $2^{17}/2^2 = 2^{15}$ groupes
5 bits d'adresse dans bloc (offset).
11 bits d'étiquette. ($31 - 15 - 5$)

1.2 Donner les différentes parties d'une ligne (bloc) de cache (nombre de bits des différentes parties). Quel est le surcoût lié aux bits de contrôle et d'étiquette (par rapport à la partie « données » du cache)

Rappels :

- l'étiquette est la même que celle de la question précédente.

- il y a 1 ou 2 bits de contrôles : 1 bits de validité (la données est-elle utilisée actuellement ?) et 1 bit de modification en case de réécriture (la donnée a-t-elle été modifiée sans avoir été modifiée aussi en mémoire principale?).
- les lignes de données sont celle de la questions précédente.

1. Étiquette : 10 bits
Contrôle : 1 bit (1 bit de validité)
Données : $16 * 8$
Surcoût : $11/139 : 7,91\%$
2. Étiquette : 9 bits
Contrôle : 2 bits (Réécriture : 1 bit valide + 1 bit modifié)
Données : $32 * 8$
Surcoût : $11/267 : 4,12\%$
3. Étiquette : 11 bits
Contrôle : 2 bits (Réécriture : 1 bit valide + 1 bit modifié)
Données : $32 * 8 = 256$ bits
Surcoût : $13/269 : 4,83\%$

Note : plus les lignes sont longues, plus les bits de contrôle sont négligeables en comparaison des données stockées. Cependant, il faut mettre ça en regard de la recopie des lignes de caches qui vont prendre plus de temps. L'espace versus le temps.

2 Cache Données

On considère une architecture possédant un cache de données de 8 Kio octets organisé en blocs de 32 octets. On considère deux cas : correspondance directe et associativité par ensembles de 2 blocs, avec le LRU (Least Recently Used) comme algorithme de remplacement (on remplace la ligne la moins récemment utilisée). Les tableaux suivants de 4096 floats (la taille d'un float est de 32 bits) sont aux adresses suivantes :

X	Y	Z	X1	Y1	X2	Y2
0x0001 0000	0x0001 4000	0x0001 8000	0x0001 C000	0x0002 0000	0x0002 4000	0x0002 8000

2.1 Quels sont les éléments des tableaux X et Y qui peuvent occuper le premier mot du premier bloc du cache ?

Correspondance directe. : $X[0]$, $X[2048]$ et $Y[0]$, $Y[2048]$

L'adresse de X est 0x0001 0000, soit en binaire 0000 0000 0000 0001 000|0 0000 000|0 0000. Sachant que l'offset est de 5 et l'index de 8 (faites les calculs pour vous entraîner), on observe que l'offset et l'index sont remplis de 0 $\Rightarrow X[0]$ est le premier mot, sur la première ligne.

S'il y a une autre valeur de X à cet endroit, il aura nécessairement que des 0 en index et offset. La prochaine adresse est 0000 0000 0000 0001 001|0 0000 000|0 0000 (soit 0x0001 2000), soit 2000_{16} octets après le début de X. $2000_{16} = 8192_{10}$, soit le $8192/4 = 2048^e$ élément du tableau. On observe que Y est 2000_{16} octets plus loin donc les mêmes calculs sont valables.

Associativité par 2 : $X[0]$, $X[1024]$, $X[2048]$, $X[3072]$ et $Y[0]$, $Y[1024]$, $Y[2048]$, $Y[3072]$

Les mêmes calculs que précédemment sont valables, mais avec 2 fois moins de bits d'index. A noter que ces valeurs peuvent aussi occuper le premier mot de la deuxième ligne selon si la première ligne est déjà occupée ou non.