

TD n° 7 : CACHES

1. Etiquettes et index de cache.

Un processeur a 2 Go de mémoire principe.

Pour les différents caches ci-dessous, on demande

- a) Quelle est la décomposition d'une adresse mémoire (figure 1) ? Donner le nombre de bits pour les parties étiquettes, index et adresse dans le bloc.
- b) Donner les différentes parties d'une ligne (bloc) de cache (figure 2). Combien y a-t-il de bits pour le contrôle, l'étiquette et la partie donnée ? Quel est le nombre total de bits du cache ? Par rapport à la partie « données » du cache, quel est le surcoût lié aux bits de contrôle et d'étiquette ?

Etiquette	Index	Adresse dans bloc
-----------	-------	-------------------

Figure 1 : décomposition d'une adresse mémoire

Etiquette	ctl	Instructions ou données
-----------	-----	-------------------------

Figure 2 : ligne de cache

- A) Cache de 2 Mo à correspondance directe et écriture simultanée avec des blocs de 16 octets
- B) Cache de 4 Mo à correspondance directe, réécriture et blocs de 32 octets.
- C) Cache de 4 Mo associatif 4 voies (4 blocs par ensemble), réécriture et blocs de 32 octets.

2. Caches données

On considère une architecture possédant un cache de données de 8K octets organisé en blocs de 32 octets. Les exercices suivants seront traités dans 2 cas : correspondance directe et associativité par ensembles de 2 blocs, avec pseudo-LRU. On considère des tableaux de 4096 flottants simple précision (32 bits), implantés aux adresses suivantes :

X	Y	Z	X1	Y1	X2	Y2
1 0000 _H	1 4000 _H	1 8000 _H	1 C000 _H	1 E000 _H	2 0000 _H	2 4000 _H

- a) Quels sont les éléments des tableaux X et Y qui peuvent occuper le mot 0 du bloc 0 du cache ?
- b) Combien de défauts de cache de données par itération interviennent dans chacune des boucles suivantes, où on suppose que les variables scalaires sont toujours en registre :

b1 for (i=0; i<N; i++) S += X[i]*Y[i];	b2 for (i=0; i<N; i++){ S1 += X1[i]*Y1[i]; S2 += X2[i]*Y2[i]; }	b3 for (i=0; i<N; i++) S1 += X1[i]*Y1[i]; for (i=0; i<N; i++) S2 += X2[i]*Y2[i];	b4 for (i=0; i<N; i++) { S1 += X[i]*Y[i]; S2 += X[i]*Z[i]; }
-----------------------------------------------------	------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------

3. Caches instructions

Un processeur a un jeu d'instructions RISC, avec des instructions de longueur fixe d'un mot. Il a un cache instructions de 2 Kmots, avec des blocs de 8 mots. Il utilise la correspondance directe. Il exécute le programme Figure 3, constitué de deux boucles imbriquées. Les seuls branchements du programme sont les deux branchements de boucle, aux adresses 239 et 1200.

Le temps pour un succès cache est T et un défaut de cache coûte 8T.

a) En négligeant l'effet des défauts de caches pour les données, quel est le temps d'exécution du programme de la Figure 3.

b) Reprendre la question précédente en supposant un cache de 1 Kmots avec correspondance directe, puis l'associativité 2 voies (2 blocs par ensemble)

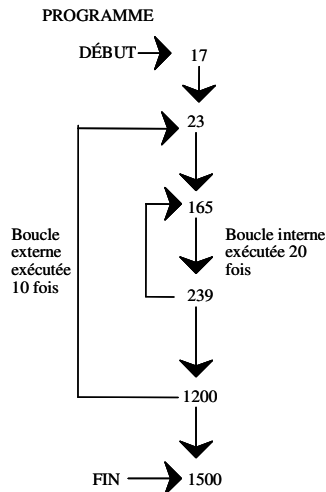


Figure 3 : programme considéré

4. Cache données

Soit le programme suivant, qui effectue la normalisation des colonnes d'une matrice X[8][8] : chaque élément de la colonne est divisé par la moyenne des valeurs de cette colonne.

```

float X[8][8], sum, ave;
sum = 0.0;
for (j = 0; j < 8; j++) {
    for (i=0; i<8, i++)
        sum+= X[i][j];
    ave = sum/8;
    for (k=7; k>=0; k--)
        X[k][j]/=ave }
    
```

On suppose que l'on a un cache de 128 octets avec des blocs de 16 octets (soit 8 blocs pour le cache). L'adresse de X[0][0] est F000_H (sur 16 bits)

- En supposant la correspondance directe, définir dans quels blocs du cache vont chaque élément de la matrice. En déduire le nombre de défauts de cache pour l'exécution du programme ? Quel serait le nombre de défauts de cache en écrivant la seconde boucle interne sous la forme : `for (k=0; k<8, k++)`
- Avec un cache totalement associatif, quel est le nombre de défauts de cache pour le programme initial en utilisant le LRU comme algorithme de remplacement ?

- Pour un cache associatif par ensemble 2 voies, définir dans quels ensembles vont les éléments de la matrices. Quel est le nombre de défauts de cache pour le programme initial en utilisant le LRU comme algorithme de remplacement ?

5. Cache données et algorithme de remplacement

Un ordinateur a une mémoire principale constituée de 1 Mo. Il a aussi un cache de 4 Ko associatif par ensemble, avec 4 blocs par ensemble et 64 octets par bloc.

- 1) Calculer le nombre de bits pour les champs Etiquette, Ensemble et Mot de l'adresse d'un mot de la mémoire principale
- 2) Le cache est initialement vide. Le processeur lit 4352 octets à partir des adresses 0, 1, 2...4351 dans cet ordre. Il répète ensuite cette séquence neuf fois. Si le cache est dix fois plus rapide que la mémoire principale, estimer l'accélération résultant de l'utilisation du cache en supposant que l'algorithme LRU est utilisé pour le remplacement d'un bloc.
- 3) Répéter la question 2 en supposant que l'algorithme de remplacement remplace maintenant le bloc le plus récemment utilisé.