École Polytechnique de Montréal

Département de génie informatique et génie logiciel

INF1600

Architecture de micro-ordinateurs

TP5

Soumis par:

Alexandre Touchette 1954855

Christophe Couturier 1959693

Section de lab #05

**Exercice 1**

L’exercice 1 se trouve dans le fichier tp5.c qui est joint à ce document.

**Exercice 2**

**1.**

Pour le placement direct :

2^14 octets / 2^4 octets/ligne = 2^10 lignes

2^10 lignes / 2^0 lignes/ensemble = 2^10 ensembles.

Le tag a 6 bits, l’ensemble a 10 bits et l’octet a 4 bits, car chaque ligne a 16 octets.

Pour le placement par groupe de 2 blocs :

2^14 octets / 2^4 octets/ligne = 2^10 lignes

2^10 lignes / 2^1 lignes/ensemble = 2^9 ensembles.

Le tag a donc 7 bits, l’ensemble a 9 bits et l’octet a 4 bits

Pour le placement par groupe de 4 blocs :

2^14 octets / 2^4 octets/ligne = 2^10 lignes

2^10 lignes / 2^2 lignes/ensemble = 2^8 ensembles.

Le tag a donc 8 bits, l’ensemble a 8 bits et l’octet a 4 bits

**2.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Direct** | | | | **2 blocs** | | | | **4 blocs** | | | |
| **Accès** | **Tag** | **Set** | **Hit** | **w-b** | **Tag** | **Set** | **Hit** | **w-b** | **Tag** | **Set** | **Hit** | **w-b** |
| WR 0x5EF1D | 17 | 2F1 |  |  | 2F | 0F1 |  |  | 5E | F1 |  |  |
| WR 0X19C7C | 06 | 1C7 |  |  | 0C | 1C7 |  |  | 19 | C7 |  |  |
| RD 0X5EF1B | 17 | 2F1 | x | x | 2F | 0F1 | x |  | 5E | F1 | x |  |
| RD 0X8CDB0 | 23 | 0DB |  |  | 46 | 0DB |  |  | 8C | DB |  |  |
| WR 0X3CDB3 | 0F | 0DB | x | x | 1E | 0DB | x |  | 3C | DB | x |  |
| WR 0X5EF15 | 17 | 2F1 | x | x | 2F | 0F1 | x | x | 5E | F1 | x |  |
| RD 0X68DBF | 1A | 0DB |  |  | 34 | 0DB | x | x | 68 | DB | x |  |
| WR 0XCAF1C | 32 | 2F1 |  |  | 65 | 0F1 |  |  | CA | F1 | x |  |
| RD 0X39C7E | 0E | 1C7 | x | x | 1C | 1C7 | x |  | 39 | C7 | x |  |
| WR 0XCAF1A | 32 | 2F1 | x | x | 65 | 0F1 | x | x | CA | F1 | x | x |

Exemple de calcul pour le placement direct

Tag 6 | set 10 | octet 4

L’adresse de l’accès est : 5EF1D

Voici sa représentation en binaire :

0101 1110 1111 0001 1101

Maintenant, on sépare les bits en respectant les valeurs trouvées plus tôt pour le tag, le set et l’octet.

0101 11 | 10 1111 0001 | 1101

On peut maintenant regrouper les groupes afin de respecter la lecture de gauche à droite (les groupes de 4 bits sont à droite) :

01 0111 | 10 1111 0001 | 1101

Les mêmes étapes sont suivies afin de trouver le set et le tag de tous les accès pour les trois types de placements.

**3.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Direct** | | **2 blocs** | | | **4 blocs** | | | | |
| **Set** | **Tag0** | **Set** | **Tag0** | **Tag1** | **Set** | **Tag0** | **Tag1** | **Tag2** | **Tag3** |
| 2F1 | 32\* | 0F1 | 2F\* | 65\* | F1 | CA\* | 5E | 5E | CA |
| 1C7 | 0E\* | 1C7 | 0C | 1C | C7 | 19 | 39 |  |  |
| 0DB | 1A\* | 0DB | 34\* | 1E | DB | 8C | 3C | 68 |  |

**4.**

Pour Direct :

10 accès dont 5 succès : 5\*8 + 5\*100 = 540 ns/10 accès = 54 ns / accès

Pour 2 Blocs :

10 accès dont 6 succès : 6\*8 + 4\*100 = 448 ns/10 accès = 44.8 ns / accès

Pour 4 Blocs :

10 accès dont 7 succès : 7\*8 + 3\*100 = 356 ns/10 accès = 35.6 ns / accès

**5.**

Si la politique de placement de la cache était complètement associative, l'adresse d'accès serait plus courte puisqu'elle comprendrait seulement le tag. La structure complètement associative n'utilise pas d'ensemble. Cela signifierait que chaque tag est différent, et il n’y aurait pas de problème d’accès à la mémoire.