TP3 : Architecture interne du contrôleur de cache

GONG Weiyi 28712019

C. Application logicielle Question C1 : Quelle est l'adresse de la première instruction de la fonction main ? Quelle est l'adresse de la première instruction de la boucle (instruction correspondant à l'étiquette loop) ? Je cherche sur le fichier seg.ld, je peut voir que main est dans l'adresse de code base:

```
    seg_reset_base  = 0xBFC00000;

    seg_kcode_base  = 0x80000000;
    seg_kunc_base   = 0x81000000;
    seg_kdata_base  = 0x82000000;

    seg_code_base   = 0x00400000;
    seg_data_base   = 0x01000000;
    seg_stack_base  = 0x02000000;

    seg_tty_base    = 0x90000000;
    seg_timer_base  = 0x91000000;
    seg_ioc_base    = 0x92000000;
    seg_dma_base    = 0x93000000;
    seg_gcd_base    = 0x95000000;
    seg_fb_base     = 0x96000000;
    seg_icu_base    = 0x9F000000;
```

Donc l'adresse base pour main est 0x004000000.

Question C2 : Quelle sont les adresses de base des trois tableaux A, B, C ? Tableau @A= 0x01000000+0x80=0x01000080 Tableau @B= @A+0x80=0x1000100 Tableau @C= @B+0x80=0x1000180

Question C3 : Pourquoi l'instruction sw est-elle placée après l'instruction bne (test de fin de boucle) ? Pour éviter le nop parce qu'il existe délay de slot après branchement, donc chaque fois le boucle arrive à la branchement, il existe un délay de slot, on a mis le prochain instruction après le branchement, donc on peut gagner une cycle.

Question C4 : Quel est le nombre de cycles nécessaires pour exécuter une itération de la boucle, dans l'hypothèse ou toutes les instructions et toutes les données sont dans le cache (pas de MISS).

Par compter les cycles il faut executer, je trouve 7 cycles sont occupés par le loop.

D. Fonctionnement du cache instruction

Question D1 : Sur combien de bits sont codés les trois champs BYTE, SET, et TAG d'une adresse 32 bits ? Quelles sont leurs valeurs pour l'adresse 0x00400000 (correspondant à la première instruction du programme)? On a toujour BYTE=4, on a 8 lignes, donc SET=3 pour identifier 8 bits, donc TAG=32-4-3=25.

TAG	SET	BYTE
0 0100 0000 0000 0000 0000 0000	000	0000

Question D2 : Représentez dans le tableau ci-dessous le contenu du cache instruction à la fin de la première itération de la boucle. Quelles instructions ont déclenché un MISS sur le cache instruction pour atteindre cet état ?

TAG	V	WORD3	WORD2	WORD1	WORD0
0 0100 0000 0000 0000 0000 0000	1	lw $10, 0($8)	li $6, 0	li $7, 20	la $8, A
0 0100 0000 0000 0000 0000 1000	1	add $12, $10, $11	addi $8, $8, 4	addi $6, $6, 1	lw $11, 128($8)
0 0100 0000 0000 0000 0001 0000	1	addiu $29, $29, -4	la $4, message	sw $12, 252($8)	bne $6, $7, loop
0 0100 0000 0000 0000 0010 0000	1	addi $8, $8, 4	addi $6, $6, 1	lw $11, 128($8)	lw $10, 0($8)
0 0100 0000 0000 0000 0011 0000	1	bne $6, $7, loop	add $12, $10, $11	bne $6, $7, loop	add $12, $10, $11

J'ai trouvé que pour touts les lignes, il génère un miss compulsive dans chaque première élément de première itération.

```
lw   $11, 128($8)
bne  $6, $7, loop
```

Question D3 : Quel est le contenu du cache instruction à la fin de la 20e itération ? Calculez approximativement le taux de MISS lors de l'exécution complète du programme (on ne prend pas en compte ce qui se passe pendant la séquence de boot et pendant l'affichage du message).

Pour le première itération, on peut bien trouver qu'il existe 3 miss pour trois lignes d'instructions, donc en totale, il y a 3 Miss, donc taux de Miss est 3/(20*7+3)=2.098%

Question D4 : Pour quel type de cache l'état MISS_SELECT est-il indispensable ?

Pour les caches associatives.

Question D5 : Complétez le graphe de l'automate ICACHE_FSM ci-dessous, en attachant à chaque transition sa condition de franchissement. Les conditions de transition dépendent des signaux Booléens suivants :

```
IREQ : le processeur emet une requête instruction,
IUNC : l'adresse correspond à un segment non caché,
IMISS : le cache instruction fait MISS,
VALID : l'automate PIBUS signale que la réponse est disponible,
ERROR : l'automate PIBUS signale que la réponse correspond à une erreur.
```

Signal	Fonction	Commentaire
A	IREQ and IUNC and not IMISS	S'il est dans l'état de IDLE, quand il y'a les demandes pour non cachable, après il entres UNC_WAIT.
B	IREQ and IMISS and not IUNC	S'il a des demandes pour miss, donc il va aller dans l'état de MISS_SELECT.
C	not IREQ ou (IREQ and IUNC and IMISS) ou(IREQ and not IUNC and not IMISS)	Pas de demande, donc pas de mouvement.
F	not ERROR and VALID	Quand il est dans l'état de MISS_Wait, il va attendre valider pour entrer MISS_UPDT.
G	ERROR	Le PIBUS il ne réponds pas pour séléctionner le bon cache.
H	not ERROR and VALID	Il est en train de attends pour le validation.
I	1	Après bien envoies les messages pour séléctionner le cache, il sorts l'état.
J	not ERROR and not VALID	Même condition comme H.
K	ERROR	Même condition comme G.
L	not ERROR and VALID	Même condition comme F.

Signal	Fonction	Commentaire
M	1	A la fin de la opération.
N	1	A la fin de la opération.
O	1	S'il existe les errors, il rentres dans l'état IDLE et éffaces les données avant.

**Question D6 : Dans quel état est forcé cet automate lors de l'activation du signal RESETN? Quel doit être l'autre effet du signal RESETN sur le cache instruction ? **

Dans l'état de IDLE, le signal RESETN est forcé activé pour invalider tous les instructions dans icache.

**Question E1 : Quelles sont les valeurs des trois champs BYTE, SET, et TAG pour les adresses des éléments A[0] et B[0] des deux tableaux ? Donnez le contenu du cache de données à la fin de la première itération de la boucle, en précisant quelles instructions entraînent un MISS sur le cache de données et donc un gel du processeur. **

A[0]:

TAG	SET	BYTE
0 0000 0010 0000 0000 0000 0001	000	0000
B[0]:		
TAG	SET	BYTE
-------	-------	-------
0 0000 0010 0000 0000 0000 0010	000	0000

TAG	V	WORD3	WORD2	WORD1	WORD0
0 0000 0010 0000 0000 0000 0001	1	1	2	3	4
0 0000 0010 0000 0000 0000 0010	1	101	102	103	104

On entraîne sur les deux instructions:

```
        lw   $10,  0($8)  # $10 <= A[i]
        lw   $11,  128($8)  # $11 <= B[i]
```

Question E2 : Calculez le taux de MISS sur le cache de données pour l'exécution complète des 20 itérations de cette boucle. Donnez le contenu du cache de données à la fin de la 20e itération de la boucle.

TAG	V	WORD3	WORD2	WORD1	WORD0
0 0000 0010 0000 0000 0000 0010	1	104	103	102	101
0 0000 0010 0000 0000 0000 0010	1	108	107	106	105
0 0000 0010 0000 0000 0000 0010	1	112	111	110	109
0 0000 0010 0000 0000 0000 0010	1	116	115	114	113
0 0000 0010 0000 0000 0000 0010	1	120	119	118	117
	0				
	0				

Taux de Miss=5/10=50%

Question E3 : Complétez le graphe de cet automate, en attachant à chaque transition sa condition de franchissement. Ces conditions dépendent des signaux Booléens suivants :

```
 DREQ : le processeur emet une requête de donnée,
WRITE : il s'agit d'une requête d'écriture,
DMISS : le cache de donnée fait MISS,
DUNC : l'adresse correspond à un segment non caché,
WOK : le write buffer n'est pas plein,
VALID : l'automate PIBUS signale que la réponse est disponible..
ERROR : l'automate PIBUS signale que la réponse correspond à une erreur.
```

Question E4 : Les expressions booléennes associées aux transitions de sortie de l'état WRITE_REQ sont très proches des expressions booléennes associées aux transitions de sortie de l'état IDLE. Quelle est la différence?

Signal	Fonction	Commentaire
A	DREQ and DUNC and not DMISS and not WRITE	Comme en Question D
B	DREQ and not DUNC and DMISS and not WRITE	Comme en Question D
C	not DREQ	Comme en Question D
D	DREQ and WRITE and not WOK and not DMISS and not DUNC	S'il existe des données à écrire et le buff est plaine
E	DREQ and WRITE and WOK and not DMISS and not DUNC	S'il existe des données à écrire et le buff n'est pas plaine
F	not ERROR and VALID	Comme en Question D
G	ERROR	
H	not ERROR and not VALID	Comme en Question D
I	1	
J	not ERROR and not VALID	Comme en Question D
K	ERROR	Comme en Question D
L	not ERROR and VALID	Comme en Question D
M	1	Comme en Question D
N	1	Comme en Question D
O	1	Comme en Question D

Signal	Fonction	Commentaire
F	wok	Attends pour le buff n'est pas vide

Commentaire:

F. Accès au PIBUS

Question F1 : Pourquoi les écritures ont-elles la priorité la plus élevée ? Quel est l'inconvénient de ce choix ?

1.Pour vider le buffer plus vite possible. Pour éviter le bloçage de buffer. 2.Pour vite fait le écriture pour diminuer le taux de Miss. 3.On a le mécanisme pour attendre le lecture, mais pour l'écriture, il ne peut pas attendre.

Question F2 : Quel est le mécanisme utilisé par les deux automates ICACHE_FSM et DCACHE_FSM pour transmettre une requête de lecture vers l'automate PIBUS_FSM ? Comment le serveur signale-t-il aux clients que la requête a été prise en compte ? Dans le cas d'une requête de lecture, comment le serveur signale-t-il au client que les données sont disponibles ?

Les deux automates travaillent comme: 1.Après reçu le demande pour récuperer les données, les deux automates vont chercher dans les caches, s'il exists les données, ils vont renvoyer les données directement. 2.S'il n'exist pas dans le cache, les deux automates vont envoyer les demandes vers PIBUS pour lire les données dans la mémoire. 3.S'il exist les données dans la mémoire, les deux automates vont lire les données et écrire les données dans les caches et renvoie dans le processeur vers le pi-bus.

Question F3 : Pourquoi l'automate PIBUS_FSM n'a-t-il pas besoin de signaler qu'une requête d'écriture transmise par le tampon d'écritures postées s'est effectivement terminée ? Quelle est l'utilité de la réponse dans le cas d'une transaction d'écriture ?

Il a pas besoin de signaler car il est conçu pour fonctionner selon un protocole de communication qui garantit que les transactions d'écriture sont bien traitées et finalisées avant de passer à la transaction suivante.

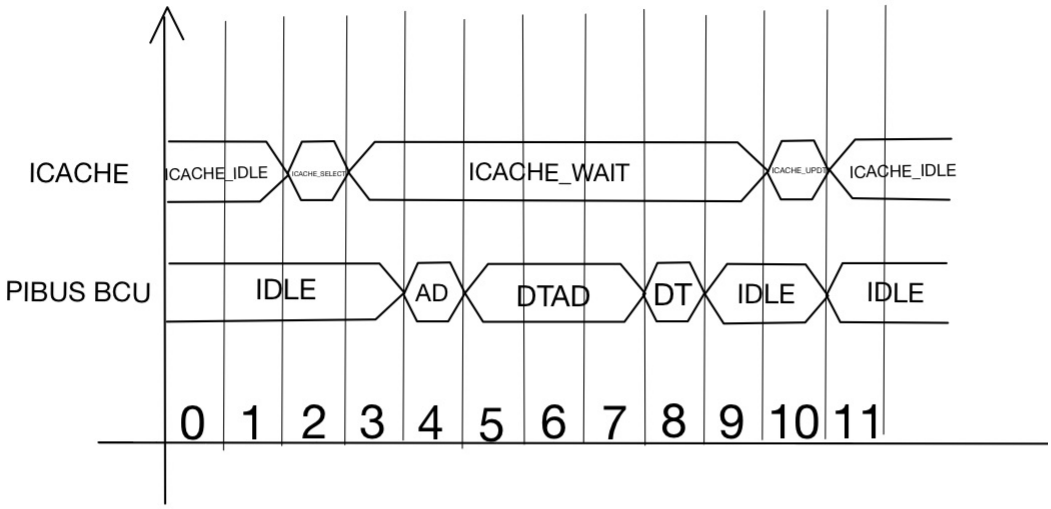
Question F4 : Complétez le graphe de l'automate PIBUS_FSM ci-dessous, en attachant à chaque transition sa condition de franchissement. Ces conditions dépendent des signaux suivants :

```
 ROK : Le tampon d'écritures postées est non vide
SC : L'automate DCACHE_FSM demande une "écriture conditionnelle"
IUNC : L'automate ICACHE_FSM demande une instruction non cachable
IMISS : L'automate ICACHE_FSM delande une ligne de cache instruction
DUNC : L'automate DCACHE_FSM demande une donnée non cachable
DMISS : L'automate DCACHE_FSM demande une ligne de cache de donnée
GNT : Le bus est alloué au composant PIbusMips32Xcache
LAST : un compteur initialisé dans l'état IDLE signale qu'il s'agit de la dernière adresse d'une rafale
ACK : Réponse de la cible (3 valeurs possibles : READY, WAIT, ERROR)
```

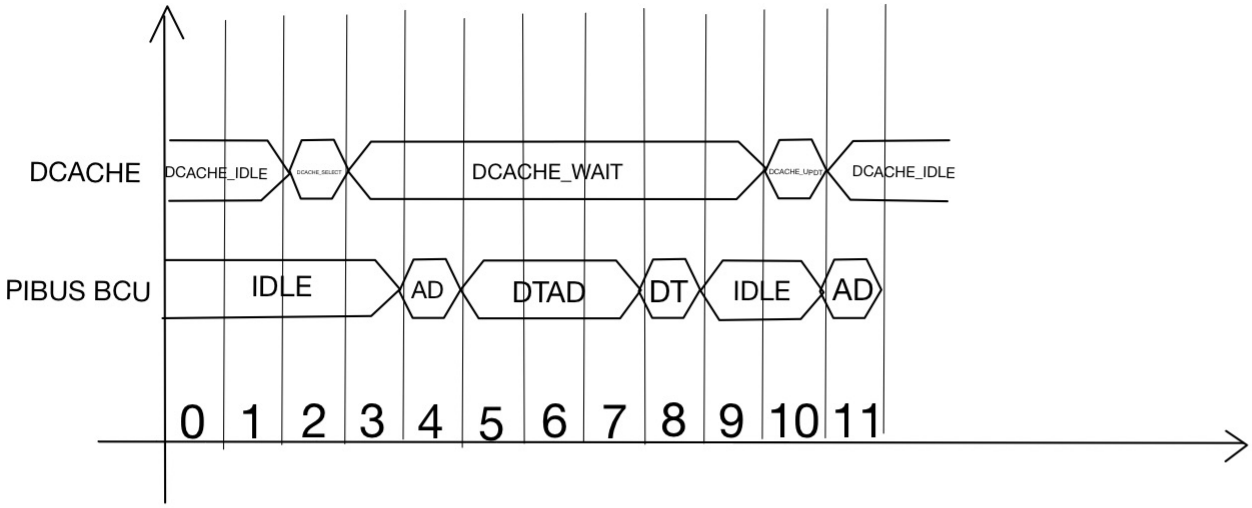
Signal	Fonction
B	GNT
B'	not GNT
C	1
D	ACK or (READY and ERROR) or LAST
D'	not LAST or (ACK and WAIT)
E	GNT
E'	not GNT
F	LAST
F'	not LAST
G	LAST
G'	not LAST
H	ACK or (READY and ERROR) or LAST
H'	not LAST or (ACK and WAIT)
X	ROK and SC
Y	not ROK and not SC or (IMISS or DMISS or IUNC or DUNC)
Z	not ROK or not SC or not IMISS or not DMISS or not DUNC or not IUNC

Question F5 : Calculez le coût minimal du miss sur le cache instruction (nombre de cycles de gel du processeur en cas de miss). Pour cela, vous dessinerez explicitement le chronogramme décrivant pour chacun des deux automates, la succession des valeurs stockées dans les registres d'etat. Même question pour le cache de données.

Graphe de ICACHE:



Graphe de DCACHE:



On peut trouver que pour les deux CACHE il toujours durée 10 cycles pour récupérent les datas, donc le MISS=10-1=9.

Question F6 : Calculez le nombre total de cycles pour exécuter les 20 itérations de la boucle de la section C, en prenant en compte les cycles de gel dûs aux miss sur le cache instruction et sur le cache de données. En déduire la valeur du CPI (nombre moyen de cycles par instruction).

Pour chaque itération, il y'a 7 instructions à traiter, 20 * 7=140 cycles. Pour chaque itération, chaque fois d'access mémoire, il a 9 cycles gel, donc pour 2 * 20 * 9=360 cycles. Au début, 4 instructions vont traités et après, les 3 miss pour instructions, ça côut 3 * 9+4=31 cycles,En totale, Miss=140+360+31=531 cycles. CPI=531/144=3.6875 cycles/instruction

G. Expérimentation par simulation

Question G1 : À quel cycle le processeur exécute-t-il sa première instruction ? A quel code correspond cette instruction ? Quel est le coût d'un MISS sur le cache instruction (coût du MISS = nombre de cycles de gel du processeur).

Le premier instruction est exécuté en cycle 57. Par consulter les data sheet: Après on a vu le fichier trace, on peut trouver que dans le cycle 115, le processeur il veut lire les données dans le RAM pour le 4 premiers données de A, jusqu'à le cycle 125, c'est le dernier fois le processeur il veut lire le dernier data dans A, donc on sait que les cycles totales pour lecture est 11 cycles, donc miss est 11-4=7 cycles.

Question G2 : À quel cycle le processeur se branche-t-il au programme principal main() ?

Dans le fichier app.bin.txt, on peut trouver que la valeur d'instruction pour le premier instruction dans main est 0x3c080100, cherches dans le fichier trace, on peut trouver que dans le cycle 50, le processeur il a récuperé l'adresse de main.

Question G3 : Quel est le coût d'un MISS sur le cache de données ? A quel cycle le processeur termine-t-il l'exécution de la première itération de la boucle ? Quelle est la durée totale de la première itération ?

Le coût d'un Miss est 6 cycles. On peut trouver que pour lire les données dans A et B, il coûte 10 cycles, donc miss=10-4=6.

Après dans le cycle 111-114, on peut voir que le processeur il appelle C[0], dans le cycle 115-125, le processeur lit les datas dans A[0]-A[3] et dans le cycle 126-141, le processeur récupère les datas dans B[0]-B[3], après le calcul, les données vont écrit dans C[1] en cycles 142-145. En totale, la première itération durée 145-111=35 cycles.

cycle 110 C[0],cycle 114 C[0],cycle 115 A[0],cycle 125 A[3],cycle 126 B[0],cycle 141 B[3],cycle 142-145 C[1],cycle 146 A[0], cycle 156 A[3], cycle 157 B[0], cycle 172 B[3], cycle 173-176 C[2], cycle 177 A[0], cycle 187 A[3], cycle 188 B[0], cycle 203 B[3], cycle 204-207 C[3],cycle 208 A[0], cycle 218 A[3], cycle 219-234 B[4]-B[7], cycle 235-249 C[5]-C[8]

Question G4 : Quelle est la durée de la seconde itération ? Quelle est la durée de la troisième itération ? Comment expliquez-vous que le coût du MISS pour les itérations 2 et 3 soit différent du coût du MISS pour la première itération ?

La deuxième itération commence en 146 et s'arrête en cycle 176, ça coûte 31 cycles. La troisième itération ça commence en 177 et s'arrête en cycle 207, ça coût 31 cycles. Pour la première itération, il faut générer le registre C, donc les 4 cycles de C sont pour générer les premières 4 words de C.

Question G5 : Quel est le taux de MISS sur le cache de données à la fin de l'exécution de la boucle ? Quelle est la durée totale du programme main (sans compter le temps d'exécution de l'affichage du message final).

Le taux de miss est 100% en théorique. On peut voir que la dernière itération s'arrête en cycle 703, donc la durée totale pour le boucle est 703-110=594.

H.Optimisation

```
A : .word   1,  2,  3,  4,  5,  6,  7,  8,  9, 10
    .word  11, 12, 13, 14, 15, 16, 17, 18, 19, 20
    .space 48

B : .word 101,102,103,104,105,106,107,108,109,110
    .word 111,112,113,114,115,116,117,118,119,120
    .space 48
```

Pour les codes ici, on peut faire avec la statégie de padding, donc il faut changer les espaces pour lui, il peut stocker dans le différent RAMs, donc ça va diminuer les miss. Chaque fois les accès mémoires vont seulement côut un seul fois de miss de nouveau ensemble, les d'autre accès mémoires vont en HIT.

```
A : .word   1,  2,  3,  4,  5,  6,  7,  8,  9, 10
    .word  11, 12, 13, 14, 15, 16, 17, 18, 19, 20
    .space 64

B : .word 101,102,103,104,105,106,107,108,109,110
    .word 111,112,113,114,115,116,117,118,119,120
    .space 64
```