centre de télé-enseignement universitaire formations à distance à votre rythme par e-learning

ctu UNIVERSITÉ DE FRANCHE-COMTÉ UFC

Licence d'informatique - Année 2017/2018

Architecture des Ordinateurs

Examen de Aout 2018 - Durée 3 heures

Documents autorisés : aucun - pas de calculatrice

- Si vous détectez une erreur de sujet, merci de l'indiquer dans votre copie.
- Pour être valables, vos réponses doivent être justifiées sur votre copie.
- Complétez et joignez la feuille de réponse à votre copie
- Utilisez les noms des variables et des fonctions proposées dans l'énoncé.
- Les questions sont indépendantes.

Exercice 1 : Représentation des nombres

```
1     void main() {
2         int N1= 0xABCD1234,N2=0x4300BCBA;
3         int N3 = Float.floatToIntBits(-16.25f); //en Java
4          //en C  x=-16.25f ; unsigned N3= *(unsigned*) & x;
5         printTabChar(toHexString(N3));
6         printTabChar(toHexString(N1&N2));
7         printTabChar(toHexString(N1|N2));
8         printTabChar(toHexString(N1^N2));
9         printTabChar(toHexString(N1>>1));
10        printTabChar(toHexString(N1>>>1));   // EN C (unsigned)N1 >>1
11        printTabChar(foo(N1));
12        printTabChar(foo(N2));
13    }
14    int foo(int x){
15        int res=0,i;
16        for (i=0;i<4;i++){
17            if ((x & 0xFF)==0)
18                res++;
19            x = x >> 8;
20         }
21        return res;
22 }
```

Question 1 : Expliquez et donnez, sur la feuille de réponse, les valeurs hexadécimales affichées par les lignes 5 à 10 du programme ci-dessus

Question 2 : Expliquez et donnez, sur la feuille de réponse, les valeurs hexadécimales affichées par les lignes 11 et 12 du programme ci-dessus. Que fait la fonction *foo* ?

Exercice 2 : Assembleur Mips

Complétez, sur la feuille de réponse, le code de la fonction *foo* de l'exercice précédent et de son appel avec la valeur N_2 . Le code l'affichera le résultat en hexadécimal.

```
# Programme utilisé pour les
    exercices suivants
# initialement $a0 = 0, $a1=32, $v0
    =0, cp = 0x200
test:
  lw  $t1,0x2000($a0)
  addi  $v0, $v0, $t1
  addi  $a0, $a0,4
  addi  $a1, $a1,-1
  bne  $a1, $zero,test
  jr  $ra
```

Exercice 3 : Hiérarchie mémoire

Soit un cache pédagogique de 256 octets, organisé par blocs de 32 où 64 octets. Nous souhaitons évaluer les performances du programme ci-dessus pour les deux tailles de bloc. Nous évaluerons la fonction de correspondance directe. La taille des adresses est de 15 bits.

Question 1 : Que fait le programme ci-dessus ? Combien d'itérations sont effectuées ?

Question 2 : Expliquez et donnez, sur la feuille de réponse, la taille en bit de l'étiquette, de l'index et du déplacement dans les deux cas.

Question 3 : Expliquez et donnez, sur la feuille de réponse, les taux de succès lors de l'accès au cache. Pour cela, complétez les valeurs hexadécimales des trois champs et du succès d'accès au cache (S ou D).

Exercice 4 : Pipeline

Question 1 : Indiquez les dépendances lecture après écriture du programme ci-dessus.

Question 2 : Expliquez ce qu'est le branchement au plus tôt.

Question 3 : Complétez, sur la feuille de réponse, le diagramme temporel.

Question 4 : Réorganisez, sur la feuille de réponse, le programme pour ne pas avoir de cycle d'attente avec l'envoi et le branchement retardé d'un cycle.

Exercice 5 : Architecture

Question 1 : Expliquez et donnez, sur la feuille de réponse, les valeurs hexadécimales représentant les instructions citées.

Question 2 : Expliquez et donnez, sur la feuille de réponse, les valeurs numériques des signaux lors du traitement des trois instructions précédentes à la première itération du programme. La valeur mémoire en **0x**2000 contient la valeur **0x**100.

Question 3 : Nous souhaitons ajouter l'instruction ***jalr rs*** qui effectue le branchement avec lien à l'adresse donnée dans *rs*. Le numéro de fonction est 9.

1. Sur le modèle de l'instruction jal expliquez comment fonctionne cette instruction ?
2. Que devons nous modifier dans le MIPS pour implémenter cette instruction ?

Exercice 6 : Exceptions et Entrées/sorties

```
.text
  lui  $t0,0x8000
  addi  $t1, $zero,1
  sub  $a0, $t0, $t1
  addi  $v0, $zero,1
  syscall
  addi  $v0, $zero,10
  syscall
```

Question 1 : Quelle valeur devrait être affichée par le programme ci-dessus (exprimée en puissance de 2)

Question 2 : Expliquez ce qui pose problème lors de l'exécution de ce programme ?

Feuille de réponse à joindre à votre copie sans nom ni numéro d'étudiant

toHexString(N3)	
toHexString(N1& N2)	
toHexString(N1 | N2)	
toHexString(N1 ^ N2)	
toHexString(N1 >> 1)	
toHexString(N1>>> 1)	
toHexString(foo(N1))	
toHexString(foo(N2))	

```
.data #debut en 0x2000
    N1 : .word 0xABCD1234, 0x4300BCBA
.text
    lw _____,_____( $zero)
    jal foo
    add $a0, $zero,_____
    addi $v0, $zero,_____
    syscall
    addi $v0, $zero,_____
    syscall
foo:
    sw $t0,0( $sp)
    addi $sp, $sp,_____
    sw _____,0( $sp)
    addi $t0, $zero,_____
    addi $v0, $zero,_____
foo_boucle:
    andi $t1, $a0,_____
    bne _____, $zero,foo_suite
    addi $v0, $v0,_____
foo_suite:
    sra $a0, $a0,_____
    addi $t0, $t0,_____
    bne _____, $zero,foo_boucle
    lw _____,0( $sp)
    addi $sp, $sp,_____
    lw _____,0( $sp)
    jr _____
```

Taille bloc	Taille Étiquette	Taille Index	Taille Déplacement
32 octets			
64 octets			

32 octets	Étiq	Index	Dép	Succès
0x2000				
0x2004				
0x2008				
0x201C				
0x203C				
0x205C				
0x207C				

Taux de succès : ——

64 octets	Étiq	Index	Dép	Succès
0x2000				
0x2004				
0x2008				
0x201C				
0x203C				
0x205C				
0x207C				

Taux de succès : ——

Inst / Cycle	1	2	3	4	5	6	7	8	9
lw *$t1*,**0x**2000(*$a0*)	LI	DI	EX	M	ER				
addi *$v0*, *$v0*,*$t1*	X								
addi *$a0*,*$a0*,4	X	X							
addi *$a1*,*$a1*,-1	X	X	X						
bne *$a1*,*$zero*,test	X	X	X	X	X				
lw *$t1*,**0x**2000(*$a0*)	X	X	X	X	X	X	X	X	

Diagramme avec envoi et branchement au plus tôt

```
# Programme réorganisé (ne pas inclure jr $ra)
 lw  $t1,0x2000($a0)
```

_____
_____
_____
_____

Instruction	Codage instruction
lw *$t1*,0x2000(*$a0*)	
add *$v0*,*$v0*,*$t1*	
bne *$a1*,*$zero*,test	

Instruction	num rs	num rt	num ec	val ec	res UAL	cp	cp suivant
lw *$t1*,0x2000(*$a0*)							
add *$v0*,*$v0*,*$t1*							
bne *$a1*,*$zero*,test							

Éléments de correction

toHexString(N3)	**0x**c182 0000
toHexString(N1& N2)	**0x**0300 1030
toHexString(N1 | N2)	**0x**ebcd bebe
toHexString(N1 ^ N2)	**0x**e8cd ae8e
toHexString(N1 >> 1)	**0x**d5e6 891a
toHexString(N1>>> 1)	**0x**55e6 891a
toHexString(foo(N1))	0
toHexString(foo(N2))	1

```
    lw   $a0,0x2004($zero)
    jal  foo
    add  $a0, $zero, $v0
    addi $v0, $zero,1
    syscall
    addi $v0, $zero,10
    syscall
foo:
    sw   $t0,0($sp)
    addi $sp, $sp,-4
    sw   $t1,0($sp)
     addi $t0, $zero,4
     addi $v0, $zero,0
```

```
foo_boucle:
    andi  $t1, $a0,0xFF
    bne   $t1, $zero,foo_suite
    addi  $v0, $v0,1
foo_suite:
    sra   $a0, $a0,8
    addi  $t0, $t0,-1
    bne   $t0, $zero,foo_boucle
   lw   $t1,0($sp)
   addi $sp, $sp,4
   lw   $t0,0($sp)
    jr   $ra
```

Taille bloc	Taille Étiquette	Taille Index	Taille Déplacement
32 octets	7	3	5
64 octets	7	2	6

32 octets	Étiq	Index	Dép	Succès
0x2000	**0x**20	0	0	D
0x2004	**0x**20	0	4	S
0x2008	**0x**20	0	8	S
0x201C	**0x**20	0	**0x**1C	S
0x203C	**0x**20	1	**0x**1C	S
0x205C	**0x**20	2	**0x**1C	S
0x207C	**0x**20	3	**0x**1C	S

Taux de succès : $\frac{7}{8}$

64 octets	Étiq	Index	Dép	Succès
0x2000	**0x**20	0	0	D
0x2004	**0x**20	0	4	S
0x2008	**0x**20	0	8	S
0x201C	**0x**20	0	**0x**1C	S
0x203C	**0x**20	0	**0x**3C	S
0x205C	**0x**20	1	**0x**1C	S
0x207C	**0x**20	1	**0x**3C	S

Taux de succès : $\frac{15}{16}$

Inst / Cycle	1	2	3	4	5	6	7	8	9
lw *$t1*,**0x**2000(*$a0*)	LI	DI	EX	M	***ER*** *$t1*				
addi *$v0*, *$v0*,*$t1*	X	LI	DI	EX	***EX*** *$t1*	M	ER		
addi *$a0*,*$a0*,4	X	X	LI	DI	DI	EX	M	ER	
addi *$a1*,*$a1*,-1	X	X	X	LI	LI	DI	EX	***M*** *$a1*	ER
bne *$a1*,*$zero*,test	X	X	X	X	X	LI	DI	***DI*** *$a1*	ER
lw *$t1*,**0x**2000(*$a0*)	X	X	X	X	X	X	X	X	LI

```
#code optimisé avec envoi et branchement retardé d'un cycle  et
    réorganisation du code
    lw   $t1,0x2000($a0)
    addi $a1, $a1,-1
    addi $v0, $v0, $t1
    bne  $a1, $zero,test
    addi $a0, $a0,4
```

Instruction	Codage instruction
lw *$t1*,0x2000(*$a0*)	**0x**8c89 2000
add *$v0*,*$v0*,*$t1*	**0x**0049 1020
bne *$a1*,*$zero*,test	**0x**14a0 fffb

Instruction	num rs	num rt	num ec	val ec	res UAL	cp	cp suivant
lw *$t1*,0x2000(*$a0*)	4	9	9	**0x**100	**0x**2000	**0x**200	**0x**204
add *$v0*,*$v0*,*$t1*	2	9	2	**0x**100	**0x**100	**0x**204	**0x**208
bne *$a1*,*$zero*,test	5	0	x	x	4	**0x**210	**0x**200