ARCHITECTURE DES ORDINATEURS
Corrigé PARTIEL Octobre 2011
Tous documents autorisés – 2H

Pour toutes les questions, on utilise le jeu d'instructions NIOS-II. Les différentes parties sont indépendantes

PARTIE 1 : Implantation mémoire

Soit la déclaration de variables suivante pour un programme C
short A,B,C ;
double D[2];
char toto[3] ;
float F,G,H,
unsigned char W ;
int X,Y,Z ;

Q 1) En supposant que A soit à l'adresse$A000\ 0000_H$, donnez les adresses (hexadécimale) de C, D[1], toto[2], W et Z. Quelle est l'occupation mémoire en octets ?

Variables	Adresse relative (decimal)	Adresse en hexa
A,B,C	0,2,4	A000 0000H, A000 0002H, **A000 0004H**
D[2]	8, 16	A000 0008H, **A000 0010H**
Toto[0] Toto [1] Toto [2]	24,25,26	A000 0018H, A000 0019H, **A000 001AH**
F,G,H	28,32,36	A000 001CH, A000 0020H, A000 00024H
W	40	**A000 0028H**
X,Y,Z	44,48,52	A000 002CH, A000 0030H, **A000 0034H**
Total	**56 octets**	

Q 2) Proposer une déclaration qui minimise l'occupation mémoire et donnez le nombre d'octets correspondant à cette nouvelle déclaration..

double D[2];
float F,G,H,
int X,Y,Z ;
short A,B,C ;
char toto[3] ;
unsigned char W ;
double = 16 octets ; float = 12 octets ; int = 12 octets ; Shorts = 6 octets ; Char =4 octets
Sans problèmes d'alignement
Total = 50 octets.

PARTIE 2 : Exécution d'instructions

Les registres du processeur NIOS contiennent les valeurs suivantes en hexadécimal.

R0	0000 0000
R1	1234 5678
R2	A000 4000
R3	FFFF FFFE
R4	ABEF CD01

Q 3) Donner le contenu des registres R5 à R10 (sous forme de huit chiffres hexadécimaux) après exécution des instructions suivantes.

a) ADD R5, R2, R1	R5= B2349678
b) SUB R6, R1, R2	R6 = 72341678
c) SLLI R7,R1, 4	R7 = 2345 6780
d) SRAI R8, R2,8	R8 = FFA0 0040
e) SRLI R9, R1, 1	R9 = 091AB3C
f) MUL R10, R1,R3	R10= DB975310

PARTIE 3 : Comparaison des jeux d'instructions ARM et NIOS II

On rappelle les modes d'adressage ARM

Mode	Assembleur	Action
Déplacement 12 bits, Pré-indexé	[Rn, #deplacement]	Adresse = Rn + déplacement
Déplacement 12 bits, Pré-indexé avec mise à jour	[Rn, #deplacement] !	Adresse = Rn + déplacement Rn ← Adresse
Déplacement 12 bits, Post-indexé	[Rn], #deplacement	Adresse = Rn Rn ← Rn + déplacement

Q 4) Donner l'instruction ou la suite d'instructions NIOS II correspondant à l'instruction ou la suite d'instructions ARM suivantes :

Instructions ARM	Instructions NIOS
a) LDR R3, [R1, #4)	a) LDW R3, 4(R1)
b) LDR R3, [R1], #4	b) LDW R3, 0(R1) ADDI R1,R1,4
c) LDR R3, [R1,#4] !	c) LDW R3, 4(R1) ADDI R1,R1,4
d)ADD R3, R1, R2 LSL # 3	d) SLLI R2,R2,3 ADD R3,R1,R2
e) CMP R1, R2 BGT Boucle	BGT R1,R2, Boucle
f) CMP R1, #4	ADDI R2,R0, 4

BEQ Boucle	BEQ R1,R2,Boucle

PARTIE 4 : Désassemblage

Soit le programme assembleur P1 qui travaille sur un tableau d'entiers T[1000] dont l'adresse de début est contenue dans le registre R1 et range le résultat dans la case mémoire RES d'adresse 00003000_H

	ADDI R2,R0, 1000
	SLLI R2,R2,2
	ADD R2,R1,R2
	ADD R3,R0,R0
	ADD R4,R0,R0
Boucle	LDW R5, 0(R1)
	LDW R6,4(R1)
	ADD R3,R3,R5
	ADD R4,R4,R6
	ADDI R1,R1,8
	BNE R1;R2, Boucle
	ADD R3,R3,R4
	STW R3, 3000_H(R0)

Q 5) Donner le programme C correspondant au programme P1

```
Int T[1000], i, RES, S0,S1 ;
S1=0; S2=0;
For (i=0 ;i<1000 ; i+=2){
        S0+=T[i] ;
        S1+=T[i+1] ;}
RES=S0+S1 ;
```

Q 6) Que fait le programme P1 ?

Le programme fait la somme des entiers du tableau d'entiers T[1000] et place le résultat dans la case mémoire d'adresse 3000H..

PARTIE 5 : Programmation assembleur

Q 7)Ecrire un programme en assembleur NIOS qui compte le nombre de valeurs d'un tableau d'entiers T[1000] qui sont supérieures ou égales à 10 et inférieures ou égales à 15.

L'adresse du tableau T est initialement contenue dans R1. On rangera le résultat dans la case mémoire d'adresse 0000 0100H

	ADDI R2,R0, 1000	1000
	SLLI R2,R2,2	4000
	ADD R2,R1,R2	Adresse après T[999]
	ADD R3,R0,R0	Accumulateur
	ADDI R4,R0,10	R4=10
	ADDI R5,R0,15	R5=15
Boucle	LDW R6, 0(R1)	
	ADDI R1,R1,4	
	BLT R6,R4, Fin	Si T[i] <10 aller à fin
	BGT R6,R5,Fin	Si T[i] > 15 aller à fin
	ADDI R3,R3,1	
Fin	BNE R1,R2,Boucle	Adresse de T[i+1] ≠ Adresse suivant celle de T[999]
	STW R3,0100H(R0)	

Q 8) Quelles modifications faut il apporter au programme de la question Q7) pour le transformer en une fonction, avec passage des paramètres par R1 (adresse de T[0] et R2 (nombre N) et retour du résultat dans R1 ?

	ADDI R2,R0, 1000	1000
PROC	SLLI R2,R2,2	4*N
	ADD R2,R1,R2	Adresse après T[N-1]
	ADD R3,R0,R0	Accumulateur
	ADDI R4,R0,10	R4=10
	ADDI R5,R0,15	R5=15
Boucle	LDW R6, 0(R1)	T[i]
	ADDI R1,R1,4	Adresse de T[i+1]
	BLT R6,R4, Fin	Si T[i] <10 aller à fin
	BGT R6,R5,Fin	Si T[i] > 15 aller à fin
	ADDI R3,R3,1	
Fin	BNE R1,R2,Boucle	Adresse de T[i+1] ≠ Adresse suivant celle de TN-1]
	ADD R1,R3,R0	Résultat dans R1
	JMP R31	

PARTIE 5 : PREDICTION DE BRANCHEMENTS

Soit un branchement qui a le comportement donné dans la table ci-dessous, P signifiant un branchement Pris et N un branchement non pris.

Branchement	P	P	N	N	P	N	P	N	P	P	N	N
Toujours pris	**P**	**P**	P	P	**P**	P	**P**	P	**P**	**P**	P	P
Prédicteur 1 bit	**P**	**P**	P	**N**	N	P	N	P	N	**P**	P	**N**
Prédicteur 2 bits	**FP**	**FP**	FP	fP	fN	fP	fN	fP	fN	**fP**	FP	fP

Q 9) Donner le nombre de bonnes prédictions de branchement en supposant

a) un prédicteur statique « toujours Pris »
b) un prédicteur dynamique 1 bit (initialisé à Pris)
c) un prédicteur dynamique 2 bits (initialisé à Fortement Pris)

a) 6 bonnes prédictions
b) 5 bonnes prédictions
c) 3 bonnes prédictions