Numéro d'anonymat

L3
X31I050

Nantes Université
2022/2023

Examen — 1re session

Durée : 2h. Le formulaire MIPS distribué en cours, ainsi qu'une feuille recto/verso de notes sont les seuls documents autorisés. Calculatrice interdite. Barème donné à titre indicatif. On donnera les réponses dans les cases prévues à cet effet et l'on rendra l'énoncé en le comptant comme un intercalaire. Toute réponse donnée en dehors de la case prévue doit être clairement annoncée sur le sujet sous peine de ne pas être prise en compte.

Exercice 1 (4 pts.)

1. *(0,5 pt.)* Expliquer pourquoi le registre PC d'une machine MIPS32 ne comporte que 30 bits ;

2. *(1 pt.)* On considère le programme MIPS de la colonne de gauche :

```
   .data
T: .word 4, 5, 6
   .text
   # [...] (code omis)
main:
   # [...] (code omis)
   li $t0, 4
   lw $a0, T($t0)
   add $t1, $t2, 6
   beq $s0, $s1, -4
```

```
addiu $8, $0, 0x00000004
lui $1, 0x00001001
addu $1, $1, $8
lw $4, 0x00000000($1)
```

   On constate que l'instruction `lw $a0, T($t0)` est remplacée par le code de la colonne de droite lors de l'assemblage. Justifier ce remplacement et expliquer le code utilisé ;

3. *(0,5 pt.)* Dans le programme MIPS de la question 1.2, on suppose que `$s0` est égal à `$s1`. Quelle sera l'instruction exécutée après « `beq $s0, $s1, -4` » ? On justifiera la réponse ;

4. *(1 pt.)* La valeur $1/20$ a t-elle une représentation finie en binaire ? Justifier ;

5. *(1 pt.)* On veut multiplier deux matrices carrées M_1 et M_2 en C++. Expliquez succinctement pourquoi il est plus efficace de faire $M_1 \times {}^tM_2$ que $M_1 \times M_2$ (rappel : ${}^tM$ correspond à la transposée de M).

Exercice 2 (5 pts.)

Dans cet exercice, tout code en langage d'assemblage MIPS écrit sans respecter les conventions vues en cours (usage des registres, respect des structures de contrôle, appels de fonctions, …) sera considéré comme invalide. On s'attachera à traduire le plus fidèlement possible le programme C++ proposé.

Traduire en langage d'assemblage MIPS la fonction C++ suivante :

```
void tri_insert(int V[], size_t n)
{
  if (n > 1) {
    tri_insert(V,n-1);
    int last = V[n-1];
    int j = n-2;
    while (j>=0 && V[j] > last) {
      shift_elements(V,j);
      --j;
    }
    V[j+1] = last;
  }
}
```

Exercice 3 (4 pts.)

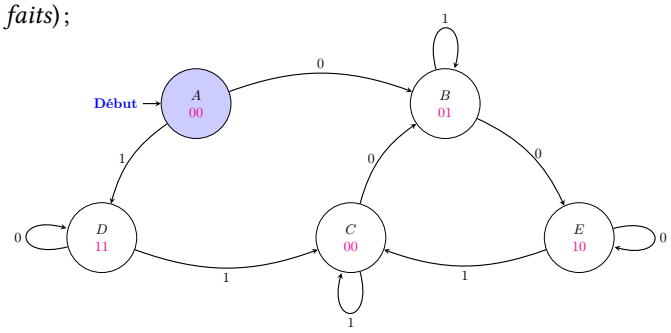
On modélise un système S avec une entrée sur un bit e et une sortie sur deux bits $s_1 s_0$ par l'automate ci-dessous.

1. *(0,5 pt.)* De quel type d'automate s'agit-il (justifier la réponse) ?

2. *(0,5 pt.)* La chaîne binaire « `011011` » est-elle reconnue par l'automate ? Si oui, quelle est la sortie produite ? (Donner aussi la succession d'états traversés) ;

3. *(2 pts.) En utilisant le codage ci-dessous*, remplissez les tableaux de KARNAUGH permettant de simplifier la fonction de transition $(q_2', q_1', q_0') = G(q_2, q_1, q_0, e)$ et la fonction de sortie $(s_1, s_0) = F(q_2, q_1, q_0)$ (*on indiquera clairement les blocs faits*) ;

États	$q_2q_1q_0$
A	000
B	001
C	010
D	011
E	100



$G(q_2, q_1, q_0, e)$

q_2q_1 \ q_0e	00	01	11	10
00				
01				
11				
10				

$F(q_2, q_1, q_0)$

q_2q_1 \ q_0	0	1
00		
01		
11		
10		

4. *(0,5 pt.)* En déduire les formes simplifiées pour s_1 et s_0 ;

5. *(0,5 pt.)* En déduire les formes simplifiées pour q_2', q_1' et q_0'.

Exercice 4 (5 pts.)

Nous nous intéressons à deux mises en œuvre différentes d'une même machine, une avec du matériel spécialisé pour le calcul flottant, l'autre sans.
Considérons un programme P ayant la répartition des opérations suivante :

multiplications flottantes	10%	divisions flottantes	5%
additions flottantes	20%	instructions entières	65%

La machine MAMF (*Machine Avec Matériel pour le Flottant*) dispose de matériel pour le flottant et peut donc mettre en œuvre directement les opérations flottantes. Pour MAMF, le nombre nécessaire de cycles d'horloge pour chaque classe d'instructions est donné par le tableau suivant :

multiplication flottante	10 cycles	division flottante	20 cycles
addition flottante	5 cycles	instruction entière	2 cycles

La machine MSMF (*Machine Sans Matériel pour le Flottant*) ne dispose pas de matériel pour le flottant et doit donc émuler les opérations flottantes avec des instructions entières. Les instructions entières prennent toutes 2 cycles d'horloge. Le nombre d'instructions entières nécessaires pour mettre en œuvre chacune des opérations flottantes est donné par le tableau suivant :

multiplication flottante	:	35 instructions entières	division flottante	:	60 instructions entières
addition flottante	:	20 instructions entières			

1. *(1 pt.)* Les machines MAMF et MSMF ont toutes les deux une fréquence d'horloge de 100 MHz. Déterminer le nombre de MIPS pour chacune des machines ;

2. *(1 pt.)* Quelle devrait être la fréquence de MSMF pour avoir des performances en termes de MIPS identiques à celles de MAMF ?

Numéro d'anonymat

3. *(1 pt.)* Si la machine MAMF a besoin de 500×10^6 instructions pour un programme P, de combien d'instructions entières a besoin la machine MSMF pour le même programme ?

4. *(2 pts.)* Quel est le temps d'exécution (en secondes) pour ce programme exécuté sur MAMF et MSMF ?

Exercice 5 (2 pts.)

On considère le code C++ ci-dessous :

```
struct tab_t {
  double *buffer;
  int sz;
};
```

```
tab_t make_tab_t(const double T[], int deb, int fin)
{
  const int taille = fin-deb+1;
  double tableau[taille];
  for (int i = deb; i <= fin; ++i) {
    tableau[i-deb] = T[i];
  }
  return tab_t{tableau, taille};
}
```

Indiquez toutes les failles et erreurs potentielles de ce code si l'on considère que les paramètres deb et `fin` sont fournis par l'utilisateur.