Université de Bordeaux
Licence Informatique 2ème année

U.F. Informatique
Année 2021 – 2022

ARCHITECTURE DES ORDINATEURS

Devoir Surveillé 1

1 heure et 20 minutes
sans documents

N.B. : - Travaillez calmement. Ne bâclez pas vos réponses : il vaut mieux traiter correctement moins de questions que tout faire de travers.
- Les réponses aux questions doivent être argumentées et aussi concises que possible.
- Le barème est donné à titre indicatif.

Exercice 1 (50 points)

On considère des entiers codés sur 16 bits, interprétés comme des entiers relatifs codés en complément à deux. Chacun de ces entiers peut (et doit) s'écrire avec quatre chiffres hexadécimaux (on omettra le préfixe « 0x »).
Soient les cinq entiers suivants :

M	N	P	Q	R
5432	BABA	C0DE	7531	8642

(1.1) (10 points)
Parmi ces 5 entiers, quels sont ceux qui sont négatifs ? Expliquez pourquoi.

(1.2) (10 points)
Classez ces entiers du plus petit au plus grand (par exemple : « M < N < P < Q < R », mais cette réponse choisie arbitrairement n'est probablement pas la bonne...). Justifiez la réponse sur votre copie.

(1.3) (10 points)
Calculez $N + Q$. Pour cela, effectuez l'addition en binaire directement sur la copie — une réponse brute ne rapportera aucun point. Y a-t-il débordement (*overflow*) ?

(1.4) (20 points)
Calculez $-Q$ et $R - Q$. Y a-t-il débordement (*overflow*) ? Comme pour la question précédente, le détail des calculs doit figurer sur la copie.

Exercice 2 (70 points)

Voici ci-dessous le texte d'un programme écrit en langage y86. Tous les commentaires ayant malencontreusement été effacés, il vous faut partir de zéro pour comprendre ce que fait ce programme, en répondant aux questions ci-dessous.
Attention : les réponses qui, au lieu d'expliquer, paraphrasent simplement le code (telles que « on soustrait %edx à %ebx ») ne rapporteront aucun point.

```
0x000:              |          .pos    0
0x000: 308400020000 | main:    irmovl  0x200,%esp
0x006: 308004010000 |          irmovl  t,%eax
0x00c: a008         |          pushl   %eax
0x00e: 8026000000   |          call    mystere
0x013: c08404000000 |          iaddl   4,%esp
0x019: 2001         |          rrmovl  %eax,%ecx
0x01b: 6000         |          addl    %eax,%eax
0x01d: 6010         |          addl    %ecx,%eax
0x01f: 400800010000 |          rmmovl  %eax,r
0x025: 10           |          halt
                    |
0x026: 502404000000 | mystere: mrmovl  4(%esp),%edx
0x02c: 308000000000 |          irmovl  0,%eax
0x032: 501200000000 | et1:     mrmovl  (%edx),%ecx
0x038: c18100000000 |          isubl   0,%ecx
0x03e: 7362000000   |          je      et3
0x043: 7255000000   |          jl      et2
0x048: 6010         |          addl    %ecx,%eax
0x04a: c08204000000 |          iaddl   4,%edx
0x050: 7032000000   |          jmp     et1
0x055: 6110         | et2:     subl    %ecx,%eax
0x057: c08204000000 |          iaddl   4,%edx
0x05d: 7032000000   |          jmp     et1
0x062: 90           | et3:     ret
                    |
0x100:              |          .pos    0x100
0x100: 00000000     | r:       .long   0
0x104: 05000000     | t:       .long   5
0x108: feffffff     |          .long   -2
0x10c: 01000000     |          .long   1
0x110: fcffffff     |          .long   -4
0x114: 00000000     |          .long   0
```

(2.1) (10 points)

Quels sont les paramètres de la fonction `mystere` ? Expliquez quelle partie du code permet de répondre à cette question.

(2.2) (20 points)

Dans la fonction `mystere`, quelles sont les rôles respectifs des registres `%eax`, `%ecx` et `%edx` ? Justifiez vos réponses.

(2.3) (10 points)

Dites, en une phrase, ce que renvoie la fonction `mystere`.

(2.4) (20 points)

Quelle est la valeur qui sera stockée à l'adresse `0x100` à la fin de l'exécution du programme ? À quoi correspond-elle par rapport à la valeur retournée par la fonction `mystere` ?

(2.5) (10 points)

Quelles sont les valeurs numériques codant les registres `%eax`, `%ecx`, `%edx` et `%esp` dans le code machine Y86 ? Justifiez vos réponses en indiquant ce qui vous permet de l'affirmer dans le code machine en colonne de gauche.

Exercice 3 (80 points)

On veut écrire une fonction Y86 respectant toutes les conventions de passage de paramètres. Cette fonction possède deux paramètres entiers `p1` et `p2`, et doit réserver la place pour trois variables locales entières `vl1`, `vl2` et `vl3`. Pour ses calculs, elle utilise les registres `%eax`, `%ecx`, `%esi` et `%edi`.

(3.1) (20 points)

Écrivez le code `y86` d'entame à placer au début de la fonction afin qu'elle soit conforme à la spécification ci-dessus. Dessinez l'état de la pile au moment de l'appel, et après l'exécution de votre code d'entame.

(3.2) (20 points)

Écrivez le code `y86` de terminaison devant être exécuté à la fin de la fonction, conduisant au retour à la fonction appelante, et correspondant à votre code d'entame.

(3.3) (10 points)

Écrivez l'instruction permettant, au sein de la fonction, de placer le premier paramètre de la fonction dans le registre `%eax`. Justifiez votre réponse.

(3.4) (10 points)

Écrivez l'instruction permettant de sauvegarder la valeur du registre `%ecx` dans la deuxième variable locale. Justifiez votre réponse.

(3.5) (10 points)

Cette fonction doit lire dans `%eax` la $i^{\text{ème}}$ ($i \geq 0$) valeur entière d'un tableau `t` déclaré en mémoire globale, la valeur de i étant elle-même stockée temporairement dans le registre `%eax` (cette valeur peut donc être détruite). Écrivez le code Y86 correspondant.

(3.6) (10 points)

Quel code écrire si l'adresse de début du tableau en question n'est pas définie en mémoire globale mais contenue dans le registre `%esi` (qui ne doit pas être détruit) ?