UNIVERSITÉ DE PARIS-DAUPHINE

CYCLE

ANNÉE : 2023 SESSION : janvier 2023

MATIÈRE : Architecture des ordinateurs

UV = ----------------

Il est interdit aussi bien de signer à la fin de la composition que d'indiquer son nom ou son numéro sur les feuilles intercalaires.

N° de groupe : TD5

Nombre d'intercalaires : 3

	Note	Signature	Note finale	APPRÉCIATIONS EXPLIQUANT LA NOTE
1er correcteur			19,25	
2e correcteur				

Ne pas écrire dans cette marge

Sujet : Exercice 1 :

1) • $8 = + 1,0 \cdot 2^3$

Donc on peut représenter 8 : 0 1010 00000

• $9 = +1,125 \cdot 2^3$

Donc 9 : 0 1010 00100

• $72 = +1,125 \cdot 2^6$

Donc 72 : 0 1101 00100

2) - Calculons 9.0 - 8.0 :

$$\begin{array}{r} 1,00100 \\ -\ 1,00000 \\ \hline 0,00100 \end{array}$$

Donc on a $9.0 - 8.0 = 0,00100_2 \cdot 2^3$

Soit $= 1,00000_2 \cdot 2^0$

$= 1.0$

Avis important : Tout candidat convaincu de fraude sera immédiatement expulsé et perdra les droits versés sans préjudice des peines prévues par le régime disciplinaire des Facultés, décret du 21 juillet 1897 (art. 33, 34 et 41) et par la loi du 23 décembre 1961.

PARTIE
A
RABATTRE

Calculons $9.0 * 1.0$

- $e = 3+0 = 3$
- $$\begin{array}{r} 1,00100 \\ \times\ 1,00000 \\ \hline 1,00100\,0.00000 \end{array}$$

Donc $9.0 * (f - 8.0)$
$= 1,00100_2 \cdot 2^3$
$= 9.0$

Si on répète l'opération une deuxième fois, on s'aperçoit bien que $f = 9.0$ car les représentations en virgule flottante sont restées exactes.

- $f = 9.0 * f - 72.0$

~~Calculons $f - 72.0$: $f = 9.0 = 1,00100_2 \cdot 2^3$~~
~~$= 0,001001_2 \cdot 2^6$~~
~~qui est arrondi à $0,00100_2 \cdot 2^6$~~
~~Suite au calcul, on obtient $f - 72.0 = -1,00000_2 \cdot 2^6$~~
~~$= -64$~~

(32)

Calculons $9.0 * f$: • $e = 3+3 = 6$

- calcul de la mantisse :

$$\begin{array}{r} 1,00100 \\ \times 1,00100 \\ \hline 100100000 \\ +\ 1001000000000 \\ \hline 1010001 0000 \end{array}$$

On a $9.0 * f = 1,010001_2 \cdot 2^6$
Arrondi à $1,01000_2 \cdot 2^6$
Ce qui est égal à 80
80 : 01101010000

Calculons $80.0 - 72.0$:

$$\begin{array}{r} 1,01000 \\ -\ 1,00100 \\ \hline 0,00100 \end{array}$$

Donc c'est égal à $0,00100_2 \cdot 2^6$

Soit $1,00000 \cdot 2^3 = 8$ (suite page 8)

Exercice 5 :

1)

instruction	temps	bloc → 1e boucle
13-14	M+C	13-14 → bloc1
15-16	M+C	15-16 → bloc2
17-18	M+C	17-18 → bloc3
1-2	M+C	1-2 → bloc4
3-4	M+C	3-4 → bloc5
5-6	M+C	5-6 → bloc6
13-14	2C	bloc1
15-16	2C	bloc2
17-18	2C	bloc3
7-8	M+C	7-8 → bloc4
9-10	M+C	9-10 → bloc5
11-12	M+C	11-12 → bloc6 ⇒ T = 9M + 15C

2e boucle :

instruction	temps	bloc
13-14	2C	bloc1
15-16	2C	bloc2
17-18	2C	bloc3
1-2	M+C	1-2 → bloc 4
3-4	M+C	3-4 → bloc5
5-6	M+C	5-6 → bloc6
13-14	2C	1
15-16	2C	2
17-18	2C	3
7-8	M+C	7-8 → bloc 4
9-10	M+C	9-10 → bloc5
11-12	M+C	11-12 → bloc6 ⇒ T = 6M + 18C

les boucles 3, 4 et 5 prennent le même temps d'exécution que la boucle 2. On en conclut :

$T_{tot} = 9M + 15C + 4(6M + 18C)$
$= 33M + 87C$

2) 1e boucle :

instruction	temps	bloc
13-16	M+3C	13-16 → 1
17-18	M+C	17-20 → 2
1-4	M+3C	1-4 → 3
5-6	M+C	5-8 → 1
13-16	M+3C	13-16 → 2
17-18	M+C	17-20 → 3
7-8	2C	1
9-12	M+3C	9-12 → 2

→ $T = 7M + 17C$

2e boucle

instruction	temps	bloc
13-16	M+3C	13-16 → 3
17-18	M+C	17-20 → 1
1-4	M+3C	1-4 → 2
5-6	M+C	5-8 → 3
13-16	M+3C	13-16 → 1
17-18	M+C	17-20 → 2
7-8	2C	3
9-12	M+3C	9-12 → 1

→ $T = 7M + 17C$

On en déduit que $T_{tot} = 5(7M + 17C) = 35M + 85C$ car encore une fois, le temps de la 2e boucle est égal à celui de la 3e, 4e et 5e.

4/9

3) 1e boucle :

instructions	temps	bloc
13-18	$M+5C$	13-18 → 1
1-6	$M+5C$	1-6 → 2
13-18	$6C$	1
7-12	$M+5C$	7-12 → 2

$\Rightarrow T_1 = 3M + 21C$

2e boucle :

13-18	$6C$	b1
1-6	$M+5C$	1-6 → b2
13-18	$6C$	b1
7-12	$M+5C$	7-12 → b2

$\Rightarrow T_2 = 2M + 22C$

Donc $T_{tot} = 3M + 21C + 4(2M+22C) = 11M + 109C$

4) 1e boucle :

13-18	$M+5C$	13-24 → b1
1-6	$M+5C$	1-12 → b1
13-18	$M+5C$	13-24 → b1
7-12	$M+5C$	7-12 → b1

$\Rightarrow T = 4M + 20C$

2e boucle :

13-18	$M+5C$	13-24 → b1
1-6	$M+5C$	1-12 → b1
13-18	$M+5C$	13-24 → b1
7-12	$M+5C$	1-12 → b1

$\Rightarrow T = 4M + 20C$

Donc $T_{tot} = 5(4M+20) = 20M + 100C$

5) On ne peut pas contrôler nous-même à quelles adresses mémoires vont les instructions. Ainsi on ne pourra jamais optimiser l'efficacité du cache.
Cependant on remarque que ce cache LRU semble être le plus efficace avec plusieurs blocs contenant un nombre élevé d'instructions (le cache est maximisé ici pour 2 blocs de 6 instructions.

Exercice 2

1)

a_0	s_0	r_0
0	1	0
1	0	1

$s_0 = \bar{a}_0$ et $r_0 = a_0$

2)

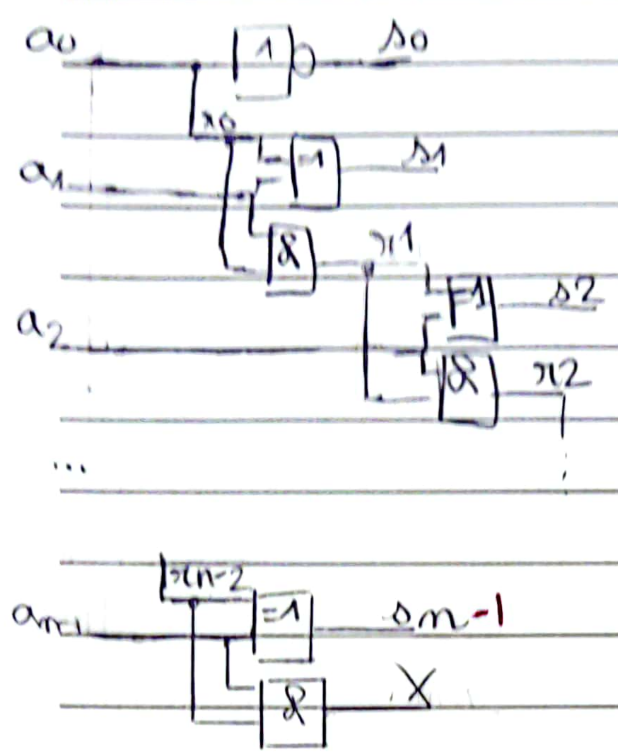
a_i	r_{i-1}	s_i	r_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$s_i = a_i \oplus r_{i-1}$
$r_i = a_i r_{i-1}$

3) On peut représenter avec A des nombres allant de 0 à $2^n - 1$.
Or S représente A+1. $2^n - 1 + 1 = 2^n$ ce qui n'est pas représentable. Donc $A = 111\ldots1_2 = X$
On a donc $X = r_{n-1}$
car la retenue finale n'est égale à 1 que si tous les a_i sont égaux à 1.
5) On a $r_{n-1} = a_{n-1} r_{n-2} = a_{n-1} a_{n-2} r_{n-3}$
$= \ldots = \prod_{i=1}^{n-1} a_i \, r_0 = \prod_{i=1}^{n-1} a_i$

$S_i = ?$

4)



Exercice 3 :

```
      MVi r1, #1          attention au cas r0=0|1   // Cas de base : r0 est premier
      MVi r2, #2                                    // le diviseur (nombre < r0)
loop: SUB r31, r0, r2                               // tant que le diviseur est différent
      JZ r31, fin                                   de r0 on fait la boucle
      DIV r3, r0, r2                                // On calcule la division entière de
                                                    r0 par r2
      MUL r3, r3, r2
      SUB r4, r0, r3                                // a=qb+r donc r = a-qb
      JNZ r4, suite                                 // si le reste==0, alors le nombre
      MVi, r1, #0                                   n'est pas premier
      JMP fin
suite: ADD r2, r2, #1
      JMP loop
fin:
```

Exercice 4 :

```
        MVi r3,#0                //passer 10 fois dans la boucle
loop1:  SUB r31,r3,#10           à l'aide du compteur r3
        JZ r31,fin
        MVi r5,#0                //compte les occurences dans le tableau de r3
        MVi r6,#0                //il faut passer 20 fois dans la boucle
loop2:  SUB r30,r6,#20           (on s'aide du compteur r6)
        JZ r30,suite1
        LDB r2,(r0)
        SUB r29,r2,r3            //si r2==r3, alors il y a une
        JNZ r29,suite2           occurence de r3 dans le tableau
        ADD r5,r5,#1             donc on augmente la valeur de r5
suite2: ADD r0,r0,#1
        ADD r6,r6,#1
        JMP loop2
suite1: STB (r1),r5              //on met le nombre d'occurences (r5) dans le tableau (r1)
        SUB r0,r0,#20            //pour que le registre contienne bien le 1er
        ADD r3,r3,#1             élément du tableau, on enlève la taille du
        ADD r1,r1,#1             tableau à r0.
        JMP loop1
fin :   SUB r1,r1,#10            //r1 pointe sur la 10e case donc on
                                 le remet à la 1e case du tableau
```

Exercice 1 (suite) :

On réeffectue la boucle : $9.0*f$:

$$\begin{array}{r} 1,00100 \\ \times 1,00000 \\ \hline 1,00100\,00000 \end{array}$$

et $e = 3+3 = 6$

Donc $9.0*f = 1,00100_2 \cdot 2^6$
$= 72.0$

Ainsi $72.0 - 72.0 = 0.0$

Donc après la boucle on aura $f = 0.0$.

- L'arrondi s'effectue désormais supérieurement.
donc à la fin du premier passage de la boucle on aurait $f = 10 \cdot 0$

$9 \cdot 0 * f = 1{,}01001 \cdot 2^5 = 82 \cdot 0$

$82 \cdot 0 - 72 \cdot 0 = 1{,}01000 \cdot 2^3 = 10 \cdot 0$

Après le 2e passage, on a $f = 18 \cdot 0$
On peut en déduire que plus on augmente la taille de la boucle, plus f augmente, qui finit par tendre vers l'infini